Occupation Games: A Logical Study

Thesis submitted by

Mohamed Anas Noor Mohamed
CS1915

under the guidance of

Dr. Sujata Ghosh, Indian Statistical Institute, Chennai

in partial fulfilment of the requirements
for the award of the degree of

Master of Technology

INDIAN STATISTICAL INSTITUTE KOLKATA

July 2021

DECLARATION OF AUTHORSHIP

I,**Mohamed Anas Noor Mohamed**, declare that this thesis titled **"Occupation Games: A Logical Study"** and the work presented in it, submitted to Indian Statistical Institute, Kolkata, is a bonafide record of the study carried out in partial fulfillment for the award of the degree Master of Technology in Computer Science. I confirm that:

- No part of this thesis has previously been submitted for a degree or any other qualification at this institute.
- I have acknowledged all relevant sources of help.

N. Mohamed Anas

Mohamed Anas Noor Mohamed
Date: 11th July 2021

THESIS CERTIFICATE

This is to certify that the thesis titled **Occupation Games: A Logical Study**, submitted by **Mohamed Anas Noor Mohamed (CS1915)**, to the Indian Statistical Institute Kolkata, for the award of the degree of **Master of Technology**, is a bona fide record of the research work done by him under our supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Dr. Sujata Ghosh
Associate Professor
Indian Statistical Institute,Chennai
Date: 11th July 2021

Dedicated to my Parents, Grandparents

ACKNOWLEDGEMENTS

I sincerely thank Dr.Sujata Ghosh for accepting me as her student and for being an amazing supervisor. More than anything she is an amazing teacher. The course on Logic taught by her gave me technical foundation piqued my interest in the subject. Her support and motivation helped a lot when I was having lack of ideas. Her critical feedback on my ideas immensely helpful in shaping my thoughts. I am also grateful for her painstaking efforts in going through and suggesting improvements for my manuscript.

I am deeply grateful to ISI for providing with resources and also full of motivating and passionate teachers to learn from and being a source of inspiration.

Finally, I am always indebted to my parents for doing their best to help me focus on studies and for their constant source of care and love.

ABSTRACT

Graph games are played on a directed or un-directed graphs with two or more players. Each game differs from one another in the actions available for each player and the constraints present. The different constraints offer a wide variety of interesting situations to analyze. They help in modelling real life interactive situations and have a wide variety of applications. In this project we will be studying logical aspects of a graph game known as occupation game. The occupation game is a two player, turn-based game played on a graph in which each player starts in certain vertices of a un-directed graph and takes turns in moving to their adjacent vertices. Each player "poisons" the vertex they are currently on, thereby restricting their opponent's movement to that vertex in the graph. The winner is the player who reaches her goal state first. This game is an extension of the poison game, which was introduced by P. Duchet and H. Meyniel in 1993. Poison game is a two player graph game where only a single player "poisons" the vertices. This game has been well studied both graphically and in terms of a logical language, termed as Poison Modal Logic (PML). To characterize the occupation game, we develop a modal logic, called occupation logic that extends the PML in two dimensions. To the best of our knowledge, such two dimensional models have not been studied extensively before. In this project we study the model-theoretic properties and decidability of the occupation logic and also use the logic to express the winning positions in the game.

Contents

List of Figures

ABBREVIATIONS

FOL	First Order Logic
OG	Occupation Game
PML	Poison Modal Logic
OL	Occupation Logic

NOTATION

φ	A formula that follows the syntax of the language
M	A model on which formulas are evaluated $M = (W, R, V)$
W	Vertices of M
R	Edge relation of M
V	Valuation function of M
◆B	Poison operator for player A
◆A	Poison operator for player B

Chapter 1

INTRODUCTION

Games form a pastime for people of all ages. There is a rich variety of games available around us. Board games, math games are some of the examples. The advantage of analysing such games is two fold, firstly we get a good strategy of how to win the game at any point of the game, and secondly a better understanding of the tools we need to do such an analysis. Sometimes we come up with new techniques that become necessary for the investigations. Games such as Nim(https://en.wikipedia.org/wiki/Nim) have given rise to mathematical theories like Sprague-Grundy theorem [Gru64] which characterize all such 2-player games where players have the same moves in every situation. In many cases, with some modifications, games mimic certain real life situations. One such well-known game is the Prisoner's Dilemma which deals with the dynamics of social interaction between non-cooperative entities.

The field of game theory which rapidly developed in the 1950's formalized the notions of winning conditions, strategies for games between many players. Game theory is the study of mathematical models of strategic interaction among rational decision-makers [Mye91]. A strategy of a player is basically a function from the set of nodes where the player plays to the set of actions available to the player. While playing a strategy, a player considers the moves of her opponent as well. The notion of utility for a player sometimes replaces the binary win/lose condition. Utility of a player at possible end-nodes of a game can be considered as a representation of a player's preferences over outcomes. John Nash introduced the concept of Nash Equilibrium [Nas49] which talks about an equilibrium strategy profile (a tuple of strategies with one strategy for each player) of the players. Nash Equilibrium is a strategy profile with the property that no player can benefit by deviation from his corresponding strategy. The field of game theory rapidly developed since then, and optimization tools such as linear optimization, convex optimization and many others have played a significant role in the development.

Looking beyond the usual mathematical analysis, one can consider the following question: How players reason in such games? Considering any turn-based game as an example, reasoning goes like: "If I play this move, whatever the other player does, do I end up winning?". Re-framing in terms of optimization: "If I play this step, what is the minimum utility I can surely receive no matter what the other player does?". Games are not just about maximizing utilities. The players have some inherent style of reasoning and belief of how the game will proceed and such concepts lead to certain intricate study about strategizing in games. In some cases, a player can choose a move that may not be optimal but can still get a higher utility because of what he knows about the belief and reasoning of his opponent.

Logic can be considered as the formal or mathematical study of reasoning. Traditionally, it has been used to study mathematical proofs, theories and arguments of philosophy. The use of formal language to describe different concepts is very useful as we can use a common set of tools to deal with a variety of topics. Logic has its usage in computer science as well - it is extensively used in circuit checking and formal verification. We need to introduce or modify logical languages to deal with the intricacies of the subject we wish to study. Examples of such languages are propositional logic, first order logic(FOL), modal logic and others. The more complex a language is, the more expressive it is. Modal logic, a fragment of FOL is a branch of logic that is extensively used in computer science in various ways. It has certain finitary properties that put it to use in practical systems.

A game has several phases that involve reasoning/logic in various ways - deliberation prior to the game, belief revision during game-play, post game analysis [vBK20] among others. Similarly, we can also reason about structure and properties of a game. We can also explore the following questions: Given certain properties a winning state has to satisfy, can we obtain a game configuration where such properties hold; given two games that are invariant in some way, do the same winning conditions hold in both of them. Formal logic play significant role in such explorations.

This thesis concerns itself with a graph game known as Occupation Game(OG), introduced in [vBL20]. This follows the famous board game "Settlers of Catan" which allows players to occupy parts of a territory and a player cannot pass through the territory occupied by the other player. The game is played on a graph by two players A and B, say. Each player occupies a territory (in graphs, vertices) in his move. A territory occupied by player A cannot be reached by player B and vice versa. The goal for each player is to reach some vertices which are marked as goal states. The game is closely related to another graph game known as Poison Game(PG) which is well studied [DM93]. We develop a new logic, namely, Occupation Logic(OL) that extends basic modal logic with additional operators (poisoning symbols to be included here) that model the "occupation" constraints. The logic OL extends the Poison Modal Logic(PML), a logic introduced in [GR19] to reason about poison game, in terms of a two-dimensional semantics.

The thesis is structured around several chapters that introduce the context surrounding Occupation Game and the technical details and properties of Occupation Logic.

- Chapter 1 - Introduction - The current chapter gives an overview of topics, the motivation of the thesis, the use of logic and the general setting for graph games.
- Chapter 2 - Games - Introduces Occupation Game and Poison Game formally and discusses the similarities and differences between them. The characterization of Poison Games using graph-theoretic concept of kernels is also discussed

- Chapter 3 - Modal Logic - Gives a brief and quick summary of basic modal logic, the syntax, semantics and other logical properties like expressivity and decidability. Poison modal logic, which has been used to characterize winning positions in poison game is introduced. This chapter gives a skeleton of properties that we will be proving as part of the thesis.
- Chapter 4 - Occupation Logic - Introduces Occupation Logic, its syntax, semantics, formulation of winning condition of Occupation Game and develops the notions of bisimulation, expressive power, translation into First Order Logic, and others.
- Chapter 5 - Occupation Logic: Further Studies - Discusses tree model property, finite model property, decidability, and model-checking.
- Chapter 6 - Conclusion - Provides a brief summary of the topics discussed and points out future research directions.

1.1 Games on Graphs, Modal Logic

All game states can be described as configurations of the worlds, positions of players in that configuration. The act of winning is simply reaching a state satisfying certain properties. We will consider the description of game state in a few games.

Consider the popular game of chess played by two players with black pieces and white pieces. A state of the game is a tuple given as follows: (position of black pieces, position of white players, black king present, white king present). The winning condition is when either the black king or white king is removed and some other constraints within the state are satisfied. Another example is the board game Hex. Hex is played on a square-shaped hexagonal grid by two players. The players start at two adjacent sides of the square. The players alternatively place red and blue tiles anywhere on the grid with the goal being connecting the opposite sides with the same colour. A game state is a tuple given as follows: (positions of red coloured grids, positions of blue coloured grids). The winning player is the one whose coloured tiles connect the opposite sides.

Graph games are played on a directed or undirected connected graph. The general setting is that players start on different vertices of the graph. Each valid move will be to a neighbouring vertex. The winning condition may vary with players. Some examples of such conditions are reaching a specific vertex, continually moving in the graph subject to some constraints, and others. Although the above conditions look simple, what makes these games interesting are the additional constraints on the moves that can be added. Let us now look at few example games.

Sabotage game [AvBG18] is a two-player game played on a connected graph. Player A is allowed to move from a vertex to a neighbouring vertex. Player B removes edges of the graph. The goal of player A is to reach a certain vertex and the goal of player B is to stop that from happening. This game can model flows in a real world network in the presence of adversarial conditions.

A game of cops and robbers [AF84] is played on a finite connected graph. The players are C(cop) and R(robber). Initially they start on separate vertices. The moves constitute moving along an edge to a neighbouring vertex. C wins if C and R reach a common vertex (C captures R).

To reason about graph games (other games as well), it is helpful to have operators that denote the presence of moves to a neighbouring vertex and also have functions assigning possible observables to nodes of the game. They can allow for checking certain conditions, for example, whether the goal is reached, what constraints are followed by making a certain move, and others. Modal logic with its nice computational properties provides us with ways to express such situations. The $\diamond$ operator present in modal logic can be interpreted as existence of a move to a neighbouring state. The evaluation of formulas of basic modal logic is done on Kripke models $M = (W, R, V)$, where W is a set of worlds, $R = W \times W$ a relation that denotes the edges between the worlds and V is a valuation function that captures the truth values of propositional variables in the states. Here we have just given a brief overview of modal logic to give an idea of how relevant it is in modelling games, a full chapter is included in the thesis that includes and expands the above properties more formally.

Practically, the graph that a game is played on can be seen as a Kripke model. The nodes of the graph can be considered as worlds in W, the edges of the graph can be taken to be the relation R and the propositional variables are included as per the need of expressing the game observables. For example, a propositional variable called *awin* may be true at nodes where player A can win.

While the basic modal logic cannot capture all the different moves and the constraints, new operators can be introduced which model these conditions. The modelling of occupation game, which is the objective of the thesis, uses new operators to deal with the poisoning aspect of the game. Moving along, we will look at occupation games more carefully and then discuss occupation logic which has been developed to reason about them.

Chapter 2

GAMES

The main topic of discussion for this thesis is occupation game and this chapter discusses the game in details. The occupation game (OG) expands upon the mechanism introduced by poison game (PG). Thus, a discussion on PG would help us in understanding and motivating OG. PG was introduced by Duchet and Meyniel in [DM93]. The game was introduced to illustrate the concept of kernels in graphs. We provide the details below.

2.1 Poison Game

PG is a two-player game, played on a directed graph, G say. Let the players be A and B. Let V and E be the vertex set and edge set of G, respectively. The players make their move alternatively. In his move, the current player chooses a vertex $v \in V$ such that $uv \in E$ and u was chosen by their opponent in the previous move. Initially, A starts playing by selecting a vertex s. An additional condition or constraint is that after B chooses any vertex v and moves there, the vertex v is poisoned by B for A and A cannot move to v later in the game. The poisoning works only against A but not against B. B is free to move to any vertex v at any point of the game. This crucial concept of poisoning will be used in the discussion of OG. The goal of player A is to move indefinitely by avoiding the vertices poisoned by B. The goal of player B is to force A to stop $A's$ movement, i.e., A should be forced to reach a vertex such that all its adjacent vertices are poisoned. The position of both the players at each stage of the game is common knowledge. Both players know which vertices are getting poisoned as the game progresses.

It is possible to give a graph theoretic characterization of a winning strategy for player A in PG. The characterization involves the presence of kernels in a graph. A kernel is a subset of vertices in the graph that obey a set of constraints, the details of which are given below.

Definition 2.1.1 *Consider a directed graph G. Consider a set $S \subset V(G)$, where $V(G)$ is the vertex set of graph G.*

$$\Gamma_G^+(S) = \{v \in V(G) : sv \text{ is an edge from some } s \in S\}$$
$$\Gamma_G^-(S) = \{v \in V(G) : vs \text{ is an edge from some } s \in S\}$$

Local Kernel: A graph G contains a local kernel if there exists a set $S \subset V(G)$ such that S is independent (no two vertices are adjacent) and every element of $\Gamma_G^+(S)$ has a successor in S.

A local kernel is basically an independent set S of vertices of a graph such that every successor of elements in S has a successor in S itself. This means that whenever a player starts in a vertex in S, in at most two moves he can come back to an element of S. The winning condition for player A necessitates the use of a local kernel. For details, see [DM93].

To avoid dealing with certain trivial cases, the theorem stated below holds only for outwardly finite (having finite number of adjacent vertices) and progressively finite (there is no path that is infinitely long) graphs.

Theorem 2.1.2 *In a poison game played on graph G, player A has a winning strategy iff the graph G has a local-kernel and G is outwardly and progressively finite.*

Proof. The 'if' part is easier to prove. Say A starts in a vertex in local-kernel L. Then B can only poison vertices outside the local kernel. This holds because the local kernel is an independent set. So every move from local-kernel leads to a vertex outside it. Since every element of $\Gamma_G^+(S)$ has a successor in S, A can get back into local kernel and so on.
For the complete and expanded proof please check [DM93]. □

2.2 Occupation Game

Occupation game (OG) is also a two-player turn-based game played on a connected graph G. Let V,E be the vertex set and edge set of G respectively. Initially, two players, A and B, say, start at two different vertices s, t, respectively. A is now said to occupy s and B is said to occupy t. Player A starts the game. In his move, A chooses a vertex v such that $uv \in E$ and u is the current state for player A. The moves for B follow similarly. This is a crucial difference from the poison game (PG), where the moves allowed were only to the neighbouring vertices of the opponent. Additional constraint in OG is that after every move by A, the vertex he is moving to is occupied by A and similarly, after every move by B, the vertex it is moving to is occupied by B. A vertex that is once occupied by A cannot be reached by B and vice versa. At any step A can move to any adjacent vertex that is unoccupied or previously occupied by A. Similar condition holds for player B as well. Both A and B will have a separate set of vertices marked as goal states. The winner of the game is the player who is able to reach his goal state before his opponent. There might be one or more goal vertices for either player.

In this game, the current position of the players, all the goal states, vertices that are poisoned along the game-play are common knowledge. There is no hidden information between the players. The "occupation" of vertices is exactly the same as the "poisoning" of vertices in PG. It is an extended version of PG where "poisoning" can be done by both players instead of a single player. The term "poisoning" will be used henceforth to denote "occupation" of vertices.

Consider an instances of OG that happens in the graph shown in 2.1. Let the two players be A and B. A and B start at S_A and S_B, respectively and their goal states are G_A and G_B, respectively. The figure shows the effects of a series of moves in the game. The vertices shown in red are those where player A has moved to in his turns and those shown in blue are where player B has moved. The red vertices are thus poisoned by player A and cannot be reached by player B and vice versa for the blue vertices. In this instance, player B has won the game, he has reached his goal state G_B before A can reach G_A.

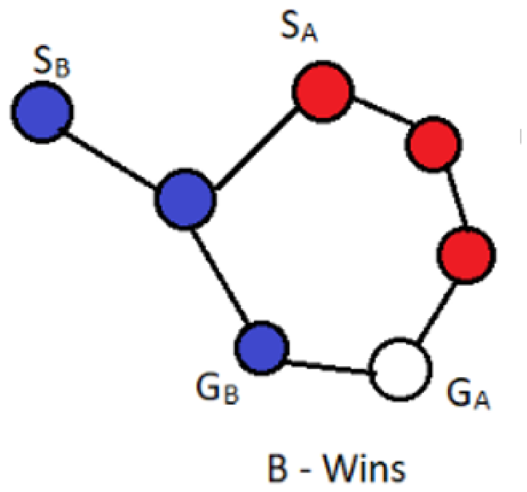

Figure 2.1: Example of an instance where player B wins

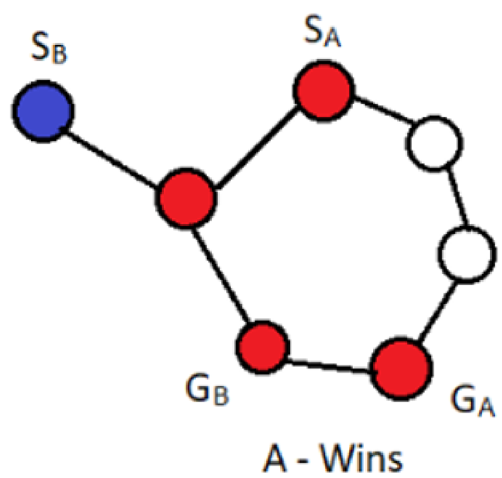

Figure 2.2: Example of instance where player A wins by obstruction

Consider another instance of the game played on the same graph shown in Fig 2.2. This time player A wins the game.

These two instances of the game highlight two different ways of winning the game. In both cases, the person who reached his goal state first wins but as the examples show, the winning strategy is not intuitive. In the game in 2.1, B wins by reaching goal state by following the shortest path from the S_B to G_B which is shorter than S_A to G_A. This illustrates a simple way of winning, the ability to poison a vertex is not made use of by either player. But the second game(2.2) highlights the important use of the ability to poison vertices. Player A's moves block all the paths for B from S_B to G_B which means B can never win the game. Now, A can easily win because none of his paths to G_A is obstructed.

Although these are simple examples, they illustrate the complex interactions brought out by the "poisoning" constraint. Given that OG is an extension of PG, there are some important differences between the two.

- Both players A and B can poison vertices in OG compared to only player B in PG.
- In PG, A starts from any vertex and B has to choose an adjacent vertex of player A and so on. This constraint is removed in OG. In OG, the starting positions of A and B are different vertices. Both A and B choose vertices adjacent to their own positions in their turn.
- The winning conditions differ as well. In PG, the winning condition for player A is to keep on moving indefinitely and that for player B is to stop that. In OG, the winning condition for each player is reaching the goal state before the other.

For PG, the existence of kernel is both necessary and sufficient condition for player A to win. In OG, with the independence of players' movements, the use of a kernel may not be that useful. In every move, the player moves to an adjacent vertex, so the movement is not restricted to an independent set. The game can sometimes end in a draw as well, where "poisoning" of vertices leads to the following situation - neither A has a path to his goal vertices nor B has a path to his. We note that the analysis of the game is done using a third-person perspective, i.e., from the viewpoint of an observer who is watching the game being played between the two players. Considering the game from the players' perspectives is another work altogether, which can be tackled in future.

Chapter 3

MODAL LOGICS

This chapter will give a brief overview of basic modal logic [BdRV10] and poison modal logic [GR19]. Some logical properties will be introduced here in the context of occupation logic (OL) which will be discussed in details in the next chapter.

3.1 Modal Logic

Modal logic historically has its origins in certain philosophical queries regarding material implication. The need to talk about more precise notions of implication such as "necessarily implies" and "possibly implies" gave rise to this logical language. The term modal comes from the phrase ‘modes of truth’. Modal logic can describe different modes of truth such as alethic, epistemic, deontic, temporal and others [Gar20]. The operators $\Box$ and $\diamond$ are introduced and used in this regard. The operators can be interpreted in many different ways based on the mode of truth we use. The symbols $\Box$ and $\diamond$ are duals of each other in a certain sense. Modal Logic is also a computationally well-behaved fragment of first order logic. The properties of this logic make it suitable for practical use in hardware and software systems which can be expressed as a labelled transition system (LTS). The graph where occupation game is being played on can also be seen as an LTS.

Definition 3.1.1 *(Labelled Transition System) A labelled transition is a tuple* (S, P, R, q_0)*, where* S *is a set of states, and* P *is a set of labels. For each label* $p \in P$*, there exists a binary relation on* S *denoted by* R_p*, and* R *is the collection of all those relations. The initial state of the system is given by* q_0*.*

3.2 Syntax and Semantics

The language of Modal logic uses propositional variables, boolean connectives along with the modal operators $\Box$ and $\diamond$. The syntax of the language L_M is defined recursively as:

$$L_M := p|\neg\varphi|\varphi \wedge \psi| \diamond \varphi$$

where $p \in P$, P being a set of propositional variables.

The formulas $\varphi \vee \psi$, $\varphi \rightarrow \psi$, $\varphi \leftrightarrow \psi$ are defined in the usual way, and $\Box\varphi$ denotes the formula $\neg \diamond \neg\varphi$. Now, given a formula in the language how do we ascertain the truth of it.

This notion is determined with respect to models of the logic which the formulas talk about. We define when a formula is "satisfied" in a model. In propositional logic, the models are given by the valuation functions which determine whether a propositional variable is true or false. In modal logic, Kripke models are used to evaluate whether a formula is true or not.

Definition 3.2.1 *(Kripke Models). A Kripke model is a tuple $M = (W, R, V)$, where W is a set of states, $R \subset W \times W$, is a binary relation on W, V is a valuation function that maps any propositional variable to a set of states (those states are said to satisfy the variable).*

Kripke models are LTS with a single label. The satisfaction of any formula is checked in a state s of a model M. The satisfaction relation of modal logic is defined recursively as

- $M, s \models p$ iff $s \in V(p), \forall p \in P$
- $M, s \models \neg\varphi$ iff $M, s \nvDash \varphi$
- $M, s \models \varphi \wedge \psi$ iff $M, s \models \varphi$ and $M, s \models \psi$
- $M, s \models \diamond\varphi$ iff there exists t s.t Rst, and $M, t \models \varphi$

One can check that the truth of $\Box\varphi$ is given by: $M, s \models \Box\varphi$ iff for all t s.t Rst, $M, t \vDash \varphi$.

3.3 Expressivity

We will now show that modal logic is a fragment of first order logic. To this end, modal logic formulas are translated to FOL formulas. The translation is defined recursively as follows

Definition 3.3.1 *(FOL Translation). Let $p_1, p_2, .. \in P$ be propositional atoms, we consider $P_1, P_2, ...$ the corresponding first order predicate symbols. We also consider a binary predicate symbol R. Let x be a free variable. $ST_x : L_M \to L$ is defined recursively as follows.*

- $ST_x(p_i) = P_i(x)$
- $ST_x(\neg\varphi) = \neg ST_x(\varphi)$
- $ST_x(\varphi \wedge \psi) = ST_x(\varphi) \wedge ST_x(\psi)$
- $ST_x(\diamond\varphi) = \exists y(R(x, y) \wedge ST_y(\varphi))$

Given a model $M = (W, R, V)$ satisfying φ at w, say, we can obtain a L-structure (M itself is considered as the corresponding L-structure) that satisfies $ST_x(\varphi)$. As mentioned above, the language L consists of unary predicate symbols P_i and a binary predicate symbol R. We construct the corresponding L-structure M_L as follows: The domain of the L-structure is W, the valuation of each unary predicate symbol P_i is $V(p_i)$ and the binary predicate symbol R is same as the relation $R \in M$. The following theorem shows the correctness of the translation.

Result 1 *For all Kriple models M and world w,*

$$M, w \models \varphi \iff M_{[x \to w]} \models ST_x(\varphi)$$

The proofs of these results are available in [BdRV10]. We will be proving these results in the context of Occupation Logic (OL).

3.4 Bisimulation

Suppose we need to establish a notion of invariance between two states a, b in models M and N respectively, which corresponds to satisfying same modal formulas - what criteria can be used to compare a and b? We may need that a and b to satisfy the same propositional variables. Also, if there is a transition from a in M, there must exist a corresponding transition from b in N.

Consider two models M, N. Bisimulation is a non-empty binary relation $Z \subset M \times N$, such that we have $(a, b) \in Z$, and for all $s \in W^M, t \in W^N$, if $(s, t) \in Z$, then they satisfy the following conditions.

- **Atom:** For any $p \in P, M, s \models p$ iff $N, t \models p$
- **Zig:** If there exists $u \in W^M$ such that $R^M su$, then there exists $v \in W^N$ such that $R^N tv$ and $(u, v) \in Z$
- **Zag:** If there exists $v \in W^N$ such that $R^N tv$, then there exists $u \in W^M$ such that $R^M su$ and $(u, v) \in Z$

Now one of the most important results related to bisimulation that has applications in variety of scenarios is the following.

Result 2 *Two bisimilar states satisfy the same modal formulas.*
Proof. The proof is by induction on the length of formula [BdRV10]. □

Bisimulation has an important application in determining which formulas are expressible in modal logic. We had seen translations from modal logic formulas to FOL formulas. The following theorem uses the concept of bisimulation to show which formulas in FOL can be obtained as translations from modal logic formulas.

Theorem 3.4.1 *(Van Benthem Characterization Theorem). Let $\varphi(x)$ be a first order formula. Then, $\varphi(x)$ is invariant under bisimulation iff $\varphi(x)$ is logically equivalent to a standard translation of a modal formula.*

3.5 Some properties

In the following we describe various logical properties related to decidability of modal logic.

3.5.1 Tree Model Property

Definition 3.5.1 *(Tree Model Property). Given any model $M = (W, R, V)$ and a world w in W, it is possible to construct a model $M' = (W', R', V')$ such that (W', R') together form a graph where a unique path exists between any two vertices(i.e., a tree) with the world w at the root of the tree, and M, w and M', w satisfy the same modal formulas.*

Modal logic has tree model property. The proof involves unravelling the all the paths from the state which we want to consider as root [BdRV10]. This property of the logic suggests that certain type of properties cannot be expressed as formulas in modal logic. For example, the presence of a cycle in a graph cannot be expressed since otherwise, a cycle has to be detected in a tree, and we arrive at a contradiction.

3.5.2 Decidability

Given any formula φ in the language, is there any model M and a world w in M that satisfy the formula? This is known as the satisfaction problem of modal logic. Decidability of a logic addresses the issue of whether the satisfaction problem of a logic is decidable, i.e., is solvable in finite time. Modal logic is decidable. This allows the use of such logic in practical systems. The proof proceeds through the use of finite model property.

Definition 3.5.2 *(Finite Model Property) Let φ be a formula. If φ is satisfiable then φ is finitely satisfiable.*

A formula is finitely satisfiable if there exist a finite model and a world in that model that satisfy it.

But a stronger bound is needed on the size of the model that satisfies the formula. If such a bound is found then a crude approach towards showing satisfiability would be to generate all models of size leading up to that upper bound and checking whether they satisfy the formula or not. This can be done in finite time.

Definition 3.5.3 *(Strong Finite Model Property) Let φ be a formula. If φ is satisfiable then φ is satisfiable in a model of size atmost $f(\mid \varphi \mid)$, where f is a computable function and $\mid \varphi \mid$ denote the number of subformulas of φ.*

Undecidability - Although modal logic is decidable, when extended with additional modal operators, as is the case with occupation logic, the decidability property is lost. Given a satisfaction problem S, to prove that S is undecidable, there should be a reduction from an undecidable problem to S. A problem that is used for these purposes is the $\mathbb{N} \times \mathbb{N}$ tiling problem.

Definition 3.5.4 *($\mathbb{N} \times \mathbb{N}$ Tiling Problem). A tile is a 1×1 square with each of its side having a colour. The $\mathbb{N} \times \mathbb{N}$ tiling problem is: Given a finite set of tile types T, is it possible to tile a $\mathbb{N} \times \mathbb{N}$ grid with tiles from T such that adjacent edges of any two tiles have the same colour.*

This tiling problem is proven to be undecidable in [Rob71]. To show undecidability, the $\mathbb{N} \times \mathbb{N}$ tiling problem is reduced to the satisfiability problem as follows. We construct a formula φ_T such that,

$$T \text{ tiles } \mathbb{N} \times \mathbb{N} \text{ iff } \varphi_T \text{ is satisfiable.}$$

3.5.3 Model Checking

In logic, model checking problem asks the following question: Given any formula φ and a model M does $M \models \varphi$ hold. It can be shown that model checking problem for modal logic is in the class P and is also $P - Hard$ [Sch02]. The latter proof involves reduction from a circuit value problem where Boolean circuits are monotone, synchronized, and properly alternating which is a $P - Complete$ problem. Monotone circuits do not use negation gates, synchronized circuits have gates organized in layers and the connections are between the layers, properly alternating means layers alternate between a layer with $\vee$ gates and a layer with $\wedge$ gates and so on.

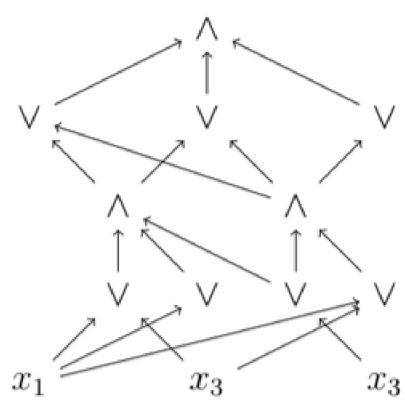

Figure 3.1: A monotone boolean circuit

Theorem 3.5.5 *The model checking problem for modal logic is $P - Hard$*

Proof. The proof involves reduction from Boolean Circuit Value problem. [Sch02]. □

3.6 Poison Modal Logic

The basic modal logic language does not have any way to express the poisoning of vertices. Since both Poison game and Occupation game make use of this constraint, a logic that extends the basic modal logic is needed. To model poison games, the language must express poisoning of a vertex and indicate that a previously non-poisoned vertex is now poisoned. Thus, certain dynamicity is involved which needs to be modelled in logic.

Poison Modal Logic or PML was introduced in [GR19] to describe reasoning in the poison game. The logic extends the basic modal logic with a poison operator $< poison >$. The syntax of the formulas in the language is defined recursively as follows:

$$L : p|\neg\varphi|\varphi \vee \psi| \diamond \varphi| < poison > \varphi,$$

where, $p \in P \cup \{p_a\}$, with P being a set of propositional variables and p_a is a propositional variable that becomes true in states which get poisoned. The $\Box$ operator is the dual of $\diamond$ operator.

As a reminder of the constraints present, every move of B poisons a vertex for A. The standard modal operator $\diamond$ talks about A's moves while the operator $< poison >$ talks about poisoning that occurs along with B's moves. The functioning of $< poison >$ operator will be clear when we look at the semantics.

The formulas are interpreted in a Kripke Model $M = (W, R, V)$, where W is same as $V(G)$, the vertex set of the graph G where the game is being played. The relation R is same as the edge set $E(G)$ of the graph G. V is a valuation function that maps a propositional variable p to a set of states S where it is true i.e. $V(p) = S$, where $S \subset V$ is the set of states where p is considered to be true. The function of the $< poison >$ operator is in adding a new state to the valuation of p_a, $V(p_a)$. The formulas are interpreted in a state w of the model M:

- $M, w \models p$ if $w \in V(p)$
- $M, w \models \neg\varphi$ if $M, w \not\models \varphi$
- $M, w \models \varphi \wedge \psi$ if $M, w \models \varphi \wedge M, w \models \psi$
- $M, w \models \diamond\varphi$ if there exists $v \in W$ such that $(w, v) \in R$ and $M, v \models \varphi$
- $M, w \models < poison > \varphi$ if there exists $v \in W$ such that $(w, v) \in R$ and $M', v \models \varphi$, where $M' = (W', R', V')$, with $W' = W, R' = R, V'(p) = V(p)$ for all $p \in P$, and $V'(p_a) = V(p_a) \cup \{v\}$.

PML can express winning positions in the game (i.e., states in the graph where players can win). For example, the winning positions of B are given as follows: Given any model M, these are states s which satisfy the formula $< poison > \Box p_a$, that is, the states where B can win in one move. The interpretation of the given formula is that there exists a "poisoning" move for B from a state s to a state t, say, from where all possible A-moves would lead to a poisoned vertex and thus A loses. This argument can be extended to winning in two moves, three moves and so on. The winning positions for player B are the states which satisfy the following infinitary formula.

$$< poison > \Box p_a \vee < poison > \Box < poison > \Box p_a \vee ...$$

The Occupation Logic (OL) introduced in the next chapter expands upon PML to meet the additional constraints of poisoning by both the players and reaching goal conditions. We now have a more symmetrical situation, so, instead of a single $< poison >$ operator we introduce two new operators $< A >$, $< B >$ (symbols will be replaced) to refer to poisoning. In PML, the evaluation was done in a single state. This was possible because the state of the game involved only the current vertex of the player but in case of OL, the game starts with players at different vertices and each move made by a player is independent of the position of his opponent. This means that a formula has to be evaluated at two states, since the current state of player A is independent of that of B. Thus we introduce the semantics of evaluating formula in a paired pointed model M, s, t. The analysis of logical properties of such two-dimensional models will be the core of Chapter 4 and Chapter 5 of this work.

This chapter has given a overview of preliminary results and properties that need to be studied for any such logic. In the subsequent chapters, we will introduce Occupation Logic and study these properties.

Chapter 4

OCCUPATION LOGIC

This chapter introduces the syntax and semantics of Occupation Logic (OL), and discusses its logical properties. The language extends the Poison Modal Logic (PML) introduced in the context of poison game (PG). Since this logic is introduced in the context of occupation game (OG), we use some terminology from the game to explain the working of the logical operators. Many proofs presented in the following chapters take ideas from [GR19].

4.1 Syntax

Two countable sets of proposition letters P_A, P_B are considered. The formulas in the language are defined recursively as

$$L_O := p_A | p_B | \neg\varphi | \varphi \wedge \psi | \Diamond_A \varphi | \Diamond_B \varphi | \blacklozenge_B \varphi | \blacklozenge_A \varphi,$$

where $p_A \in P_A, p_B \in P_B$.

We specify the poisoning of vertices by using two kinds of proposition variables, one for player A and the other for player B. The special propositional variables are $p_a, g_a \in P_A, p_b, g_b \in P_B$, where p_a is true in the states which are poisoned for A and g_a is true in states which are the goal states of player A and similarly, the other variables denote the poisoned and goal states of player B, respectively. In a game, both players can have finitely many goal states.

The $\Diamond_A$ modality talks about the adjacent states of A's current state, similarly, the $\Diamond_B$ modality talks about player B's adjacent vertices. The operator $\blacklozenge_B$is used to refer to moving to adjacent vertex and subsequent poisoning by player A (i.e., poison for player B) and the operator $\blacklozenge_A$refers to the moving and poisoning by player B.

4.2 Semantics

Given the formulas of the language, the truth of the formulas is ascertained by their interpretation on Kripke Models $M = (W, R, V)$. We use the notation, W^M, R^M, V^M to denote the (W, R, V) of a particular model M. Unlike basic modal logic and PML, where the evaluation was done at a single state of the model M, here the evaluation of a formula occurs simultaneously at two states of the model. In this case, a pointed model is a triplet (M, s, t), where $s, t \in W$. The states s, t are the current positions of player A and B, respectively. The main motivation for considering two states for evaluation is basically the independence of the positions of A and B. Let (M, s, t) be a pointed model, then

- $(M, s, t) \models p_A \iff s \in V(p_A)$
- $(M, s, t) \models p_B \iff t \in V(p_B)$
- $(M, s, t) \models \neg\varphi \iff (M, s, t) \not\models \varphi$
- $(M, s, t) \models \varphi \wedge \psi \iff (M, s, t) \models \varphi$ and $(M, s, t) \models \psi$
- $(M, s, t) \models \Diamond_A \varphi \iff \exists s' \in W, Rss'$, and $(M, s', t) \models \varphi$
- $(M, s, t) \models \Diamond_B \varphi \iff \exists t' \in W, Rtt'$, and $(M, s, t') \models \varphi$
- $(M, s, t) \models \blacklozenge_A \varphi \iff \exists t' \in W, Rtt'$, and $(M', s, t') \models \varphi$, where $M' = (W, R, V'), V'(p) = V(p) \forall p \in P, V'(p_a) = V(p_a) \cup \{t'\}$
- $(M, s, t) \models \blacklozenge_B \varphi \iff \exists s' \in W, Rss'$, and $(M', s', t) \models \varphi$, where $M' = (W, R, V'), V'(p) = V(p) \forall p \in P, V'(p_b) = V(p_b) \cup \{s'\}$

While the other satisfaction relations work in the same way as in basic modal logic, the working of $\blacklozenge_A, \blacklozenge_B$ requires some observation. The poison formula $\blacklozenge_A \varphi$ is true at (s, t) if and only if φ is true at a successor of t, t', say, with state s being the same. The pair (s, t') is considered in a model obtained from M by adding t' to the valuation of p_a. In other words, the evaluation function V changes in the sense that $V(p_a)$ changes. The notations introduced in 4.3 will help in describing such notions in a more convenient way.

With the satisfaction relation thus defined, the notion for modal equivalence in Occupation Logic is defined as follows:

Definition 4.2.1 *(Occupation Modal Equivalence). Two pointed models (M, s, t) and (N, u, v) are modally equivalent for OL if and only if for all $\varphi \in L_O$:*

$$M, s, t \models \varphi \iff N, u, v \models \varphi \qquad .$$

4.3 Introducing new notation

The evaluation of the propositional variables p_a and p_b will change as new vertices get poisoned as the game progresses. Since the valuation function is a component of model M, the model itself changes every time a new vertex gets poisoned. The following notation is used to denote the new model obtained.

Consider a pointed model $M = (W, R, V)$. Whenever player B moves to a vertex w and poisons it, we denote the newly obtained model as $M_w^{p_a}$, which is defined by,

$$M_w^{p_a} = (W, R, V'), \forall p \in P, V'(p) = V(p), V'(p_a) = V(p_a) \cup \{w\}$$

Consider a pointed model $M = (W, R, V)$. Whenever player A moves to a vertex w and poisons it, we denote the newly obtained model as $M_w^{p_b}$, which is defined by

$$M_w^{p_b} = (W, R, V'), \forall p \in P, V'(p) = V(p), V'(p_b) = V(p_b) \cup \{w\}$$

With this new notation, the satisfaction relation for $\blacklozenge_A$ is written as

$$(M, s, t) \models \blacklozenge_A \varphi \iff \exists t' \in W^M, R^M tt', \text{ and } (M_{t'}^{p_a}), s, t' \models \varphi$$

Similarly, the satisfaction relation for $\blacklozenge_B$ is written as

$$(M, s, t) \models \blacklozenge_B \varphi \iff \exists s' \in W^M, R^M ss', \text{ and } (M_{s'}^{p_b}), s', t \models \varphi$$

4.4 Validity

The following formulas are examples of validities in the logic. Given any model M and evaluation states s, t, the following formulas are satisfied by the triple M, s, t.

- All propositional tautologies
- $\boxed{A}(\varphi \to \psi) \to (\boxed{A}\varphi \to \boxed{A}\psi)$
- $\boxed{A}\bot \to \blacksquare_B \varphi$ - Since p_b is not evaluated at s.
- $\blacksquare_A p_A \to p_A, \forall p_A \in P_A$
- $\blacksquare_A p \to \boxed{B} p, \forall p \in P_B$
- $\blacksquare_A(\varphi \wedge \psi) \iff (\blacksquare_A \varphi \wedge \blacksquare_A \psi)$

4.5 Winning Strategy of Occupation Game

Considering player A moving first, the winning conditions for player A in a graph is obtained by the following idea: Player A wins if he is able to reach a goal state in the first move of the game and that state is not poisoned or if he is able to reach a non-poisoned goal state in the third move of the game and B fails to reach her goal state in second move of the game and so on. The pointed models (M, s, t) which satisfy the formula below given the starting positions (s, t) for the two players will allow the first player, player A to win.

Let $\rho_1 = \blacklozenge_B(\neg p_a \wedge g_a)$ - Winning in first move,

$\rho_2 = \blacklozenge_B(\neg p_a \wedge \neg \lozenge_B(\neg p_b \wedge g_b) \wedge (\boxed{B}((\bot \vee p_b) \to \rho_1) \wedge \neg \blacklozenge_A(\neg p_b \wedge \neg \rho_1))$ - Winning in the third move of the game.

$\rho_3 = \blacklozenge_B(\neg p_a \wedge \neg \lozenge_B(\neg p_b \wedge g_b) \wedge (\boxed{B}((\bot \vee p_b) \to \rho_2) \wedge \neg \blacklozenge_A(\neg p_b \wedge \neg \rho_2))$ - Winning in the fifth move of the game. And so on

The winning positions (s, t) for player A are defined by the following formula

$$\rho_1 \vee \rho_2 \vee \ldots$$

4.6 Translation into First-Order Logic

Let L_O denote the occupation logic language and L be the first order logic language we are translating to. Let $p, q \in P$ be propositional atoms, we call P, Q their corresponding first-order predicate. The first-order predicate for the propositions used to denote poisoning p_a, p_b are $PForA, PForB$ respectively. N, M are two finite set of variables that keep track of poisoned states for A, and poisoned states for B respectively and x, y are two designated variables. The translation $ST_{x,y}^{N,M} : L_O \to L$ is defined as follows.

- $ST_{x,y}^{N,M}(p_A) = p_A(x), \forall p_A \in P_A$
- $ST_{x,y}^{N,M}(p_B) = p_B(y), \forall p_B \in P_B$
- $ST_{x,y}^{N,M}(\neg\varphi) = \neg ST_{x,y}^{N,M}(\varphi)$
- $ST_{x,y}^{N,M}(\varphi \wedge \psi) = ST_{x,y}^{N,M}(\varphi) \wedge ST_{x,y}^{N,M}(\psi)$
- $ST_{x,y}^{N,M}(\Diamond_A\varphi) = \exists u(xRu \wedge ST_{u,y}^{N,M}(\varphi))$
- $ST_{x,y}^{N,M}(\Diamond_B\varphi) = \exists v(yRv \wedge ST_{x,v}^{N,M}(\varphi))$
- $ST_{x,y}^{N,M}(\blacklozenge_A\varphi) = \exists v(yRv \wedge ST_{x,v}^{N\cup\{v\},M}(\varphi))$
- $ST_{x,y}^{N,M}(\blacklozenge_B\varphi) = \exists u(xRu \wedge ST_{x,v}^{N,M\cup\{u\}}(\varphi))$
- $ST_{x,y}^{N,M}(p_a) = PForA(x) \vee \vee_{u\in N}(x = u)$
- $ST_{x,y}^{N,M}(p_b) = PForB(y) \vee \vee_{v\in M}(y = v)$

Similar to the case in 3.3, given a model $M = (W, R, V)$ satisfying φ at s, t, say, we can obtain a L-structure (M itself is considered as the corresponding L-structure) that satisfies $ST_{x,y}^{N,M}(\varphi)$. The FOL L consists of unary predicate symbols P_i and a binary predicate symbol R. We construct the corresponding L-structure M_L as follows: The domain of the L-structure is W, the valuation of each unary predicate symbol P_i is $V(p_i)$ and the binary predicate symbol R is same as the relation $R \in M$. The importance of the interpretation of a model M as L-structure occurs when we prove a theorem equivalent to Van Benthem Characterization theorem in modal logic 3.4.1.
The following lemmas will help with the proof of correctness of the translation.

Lemma 4.6.1 *For a model M, and an assignment g,*
$M_w^{p_a} \models ST_{x,y}^{N,M}(\varphi)[g] \iff M \models ST_{x,y}^{N\cup\{z\},M}(\varphi)[g_{z:=w}]$

Proof. We prove by applying induction on the size of the formula φ.
Base case: $\varphi = p_A, \forall p_A \in P_A$,
$M_w^{p_a} \models ST_{x,y}^{N,M}(p_A)[g] \iff M_w^{p_a} \models p_A(x)[g] \iff M \models p_A(x) \iff M \models ST_{x,y}^{N\cup\{z\},M} p_A[g_{z:=w}]$
(By the definition of poisoning and standard translation)

Base case: $\varphi = p_B, \forall p_B \in P_B \cup \{p_b\}$, Similar to p_A case
$M_w^{p_b} \models ST_{x,y}^{N,M}(p_B)[g] \iff M \models ST_{x,y}^{N\cup\{z\},M} p_B[g_{z:=w}]$

Base case: $\varphi = p_a$ - The poison proposition
$M_w^{p_a} \models ST_{x,y}^{N,M}(p_a)[g] \iff M_w^{p_a} \models PForA(x) \vee \vee_{u\in N}(x = u)[g]$
$\iff M \models PForA(x) \vee \vee_{u\in N\cup\{w\}}(x = u)[g]$
$\iff M \models ST_{x,y}^{N\cup\{z\},M}(p_a)[g_{z:=w}]$

Induction Hypothesis: The proposition is true for all formulas φ of length less than n.
To show: The proposition is true for all formulas of length n

Induction step: $\varphi = \neg\psi$
$M_w^{p_a} \models ST_{x,y}^{N,M}(\neg\psi)[g] \iff M_w^{p_a} \models \neg ST_{x,y}^{N,M}(\psi)[g]$
$\iff M \models \neg ST_{x,y}^{N\cup\{z\},M}(\psi)[g_{z:=w}]$
$\iff M \models ST_{x,y}^{N\cup\{z\},M}(\neg\psi)[g_{z:=w}]$

Induction step: $\varphi = \psi \wedge \chi$
$M_w^{p_a} \models ST_{x,y}^{N,M}(\psi \wedge \chi)[g] \iff M_w^{p_a} \models ST_{x,y}^{N,M}(\psi)[g] \wedge ST_{x,y}^{N,M}(\chi)[g]$
$\iff M \models ST_{x,y}^{N\cup\{z\},M}(\psi)[g_{z:=w}] \wedge ST_{x,y}^{N\cup\{z\},M}(\chi)[g_{z:=w}]$
$\iff M \models ST_{x,y}^{N\cup\{z\},M}(\psi \wedge \chi)[g_{z:=w}]$

Induction step: $\varphi = \Diamond_A\psi$
$M_w^{p_a} \models ST_{x,y}^{N,M}(\Diamond_A\psi)[g] \iff M_w^{p_a} \models \exists z(Rxz \wedge ST_{z,y}^{N,M}\psi)[g]$
$\iff \exists v, Rg(x)v, M_w^{p_a} \models ST_{z,y}^{N,M}(\psi)[g_{z:=v}]$
$\iff \exists v, Rg(x)v, M \models ST_{z,y}^{N\cup\{t\},M}(\psi)[g_{z:=v,t:=w}]$
$\iff M \models \exists z(Rxz \wedge ST_{z,y}^{N\cup\{t\},M}(\psi))[g_{t:=w}]$
$\iff M \models ST_{x,y}^{N\cup\{t\},M}\psi[g_{t:=w}]$

Induction step: $\varphi = \Diamond_B\psi$
$M_w^{p_a} \models ST_{x,y}^{N,M}(\Diamond_B\psi)[g] \iff M_w^{p_a} \models \exists z(Ryz \wedge ST_{x,z}^{N,M}\psi)[g]$
$\iff \exists v, Rg(y)v, M_w^{p_a} \models ST_{x,z}^{N,M}(\psi)[g_{z:=v}]$
$\iff \exists v, Rg(y)v, M \models ST_{x,z}^{N\cup\{t\},M}(\psi)[g_{z:=v,t:=w}]$
$\iff M \models \exists z(Ryz \wedge ST_{x,z}^{N\cup\{t\},M}(\psi))[g_{t:=w}]$
$\iff M \models ST_{x,y}^{N\cup\{t\},M}\psi[g_{t:=w}]$

Induction step: $\varphi = \blacklozenge_A\psi$
$M_w^{p_a} \models ST_{x,y}^{N,M}(\blacklozenge_A\psi)[g] \iff M_w^{p_a} \models \exists z(Ryz \wedge ST_{x,z}^{N\cup\{z\},M}\psi)[g]$
$\iff \exists v, Rg(y)v, M_w^{p_a} \models ST_{x,z}^{N\cup\{z\},M}(\psi)[g_{z:=v}]$
$\iff \exists v, Rg(y)v, M \models ST_{x,z}^{N\cup\{z,t\},M}(\psi)[g_{z:=v,t:=w}]$
$\iff M \models \exists z(Ryz \wedge ST_{x,z}^{N\cup\{z,t\},M}(\psi))[g_{t:=w}]$
$\iff M \models ST_{x,y}^{N\cup\{t\},M}\blacklozenge_A\psi[g_{t:=w}]$

Induction step: $\varphi = \blacklozenge_B \psi$

$M_w^{p_a} \models ST_{x,y}^{N,M}(\blacklozenge_B \psi)[g] \iff M_w^{p_a} \models \exists z(Rxz \wedge ST_{z,y}^{N,M\cup\{z\}}\psi)[g]$
$\iff \exists v, Rg(x)v, M_w^{p_a} \models ST_{z,y}^{N,M\cup\{z\}}(\psi)[g_{z:=v}]$
$\iff \exists v, Rg(x)v, M \models ST_{z,y}^{N\cup\{t\},M\cup\{z\}}(\psi)[g_{z:=v,t:=w}]$
$\iff M \models \exists z(Rxz \wedge ST_{z,y}^{N\cup\{t\},M\cup\{z\}}(\psi))[g_{t:=w}]$
$\iff M \models ST_{x,y}^{N\cup\{t\},M}\blacklozenge_B \psi[g_{t:=w}]$

This completes the proof. □

Lemma 4.6.2 *For a model M, and an assignment g,*
$M_w^{p_b} \models ST_{x,y}^{N,M}(\varphi)[g] \iff M \models ST_{x,y}^{N,M\cup\{z\}}(\varphi)[g_{z:=w}]$

Proof. The proof proceeds similar to lemma 4.6.1 □

So far, we have introduced the standard translation. As a reminder, the standard translation $ST_{x,y}^{N,M}$ is a function from L_o to L. N, M are sets that include the states that are poisoned during the evaluation of the formula, so initially, both $N = M = \emptyset$. We have to prove that the standard translation is correct. For that we need to prove the theorem below.

Theorem 4.6.3 *Let (M, s, t) be a pointed model and φ belonging to L_o be a formula in occupation logic, then:*

$$(M, s, t) \models \varphi \iff M \models ST_{x,y}^{\emptyset,\emptyset}\varphi[x := s, y := t]$$

Proof. We prove by applying induction on the size of the formula φ.
Base case: $\varphi = p_A$
$(M, s, t) \models p_A \iff s \in V(p_A) \iff M \models p_A(s) \iff M \models ST_{x,y}^{\emptyset,\emptyset}p_A(x)[x := s]$
$\iff M \models ST_{x,y}^{\emptyset,\emptyset}p_A(x)[x := s, y := t]$(Since $p(x)$ does not contain variable y)

Base case: $\varphi = p_B$
$(M, s, t) \models p_B \iff t \in V(p_B) \iff M \models p_B(t) \iff M \models ST_{x,y}^{\emptyset,\emptyset}p_B(y)[y := t]$
$\iff M \models ST_{x,y}^{\emptyset,\emptyset}p_B[x := s, y := t]$ (Since $p_B(y)$ does not contain variable x)

Now for the poison propositions
Base case: $\varphi = p_a$
$(M, s, t) \models p_a \iff M \models PForA(x)[x := s]$
$\iff M \models PForA(x)[x := s] \vee \bigvee_{z\in\emptyset} \quad z = x[x := s]$
$\iff M \models ST_{x,y}^{\emptyset,\emptyset}p_a[x := s, y := t]$

Base case: $\varphi = p_b$
$(M, s, t) \models p_b \iff M \models PForB(y)[y := t]$
$\iff M \models PForB(y)[y := t] \vee \bigvee_{z\in\emptyset} \quad z = y[y := t]$

$\iff M \models ST^{\emptyset,\emptyset}_{x,y} p_b[x := s, y := t]$

Induction Hypothesis: The above proposition we are trying to prove is true for all formulas φ of length $< n$.

To prove: The above proposition is true for formula φ of size n.

Case 1: $\varphi = \neg\psi$

$(M, s, t) \models \neg\psi \iff \textit{not} \quad (M, s, t) \models \psi$

$\iff \textit{not} \quad M \models ST^{\emptyset,\emptyset}_{x,y}\psi[x := s, y := t]$ (using Induction Hypothesis)

$\iff M \models \neg ST^{\emptyset,\emptyset}_{x,y}\psi[x := s, y := t]$

$\iff M \models ST^{\emptyset,\emptyset}_{x,y}(\neg\psi)[x := s, y := t]$

Case 2: $\varphi = \psi \wedge \chi$

$(M, s, t) \models \psi \wedge \chi \iff (M, s, t) \models \psi \wedge (M, s, t) \models \chi$

$\iff M \models ST^{\emptyset,\emptyset}_{x,y}\psi[x := s, y := t] \wedge M \models ST^{\emptyset,\emptyset}_{x,y}\chi[x := s, y := t]$ (Using Induction Hypothesis)

$\iff M \models ST^{\emptyset,\emptyset}_{x,y}(\psi \wedge \chi)[x := s, y := t]$

Case 3: $\varphi = \Diamond_A \psi$

If $(M, s, t) \models \Diamond_A \psi$, then there exists s' such that Rss' and $(M, s', t) \models \psi$, then we have that $M \models Rxu[s, s']$ and $M \models ST^{\emptyset,\emptyset}_{u,y}\psi[u := s', y := t]$.(Using Induction Hypothesis).

So $M \models Rxu[s, s', y := t]$ and $M \models ST^{\emptyset,\emptyset}_{u,y}\psi[x := s, u := s', y := t]$

This implies that $M \models (Rxu \wedge ST^{\emptyset,\emptyset}_{u,y}\psi)[x := s, u := s', y := t]$. Therefore $M \models \exists u(Rxu \wedge ST^{\emptyset,\emptyset}_{u,y}\psi)[x := s, y := t]$.

Therefore, $M \models ST^{\emptyset,\emptyset}_{x,y}\Diamond_A \psi[x := s, y := t]$

Now the other direction,

Suppose $M \models ST^{\emptyset,\emptyset}_{x,y}\Diamond_A \psi[x := s, y := t]$, then $M \models \exists u(Rxu \wedge ST^{\emptyset,\emptyset}_{u,y}\psi)[x := s, y := t]$, then $M \models (Rxu \wedge ST^{\emptyset,\emptyset}_{u,y}\psi)[x := s, y := t, u := s']$ for some s'. So we have that $M \models Rxu[x := s, u := s', y := t]$ and $(M, s', t) \models \psi$(Using Induction Hypothesis).

So there exists an s' such that Rss' and $(M, s', t) \models \psi$, therefore $(M, s, t) \models \Diamond_A \psi$.

Case 4: $\varphi = \Diamond_B \psi$. Can be proved similar to the case $\Diamond_A \psi$.

Case 5: $\varphi = \blacklozenge_B \psi$

$(M, s, t) \models \blacklozenge_B \psi \iff \exists s', Rss', (M^{p_b}_s{}', s', t) \models \psi$

$\iff \exists s', Rss', M^{p_b}_s{}' \models ST^{\emptyset,\emptyset}_{z,y}\psi[z := s', y := t]$ (Using Induction Hypothesis)

$\iff \exists s', Rss', M \models ST^{\{z\},\emptyset}_{z,y}\psi[z := s', y := t, x := s]$

$\iff M \models \exists z(Rxz \wedge ST^{\{z\},\emptyset}_{z,y}\psi)[x := s, y := t]$

$\iff M \models ST^{\emptyset,\emptyset}_{x,y}\blacklozenge_B \psi[x := s, y := t]$

Case 6: $\varphi = \blacklozenge_A \psi$. Can be proved similar to Case 5

This completes the proof. □

4.7 Occupation Bisimulation

Occupation bisimulation is the notion of invariance of pointed models with respect to occupation logic equivalence. The model/game state involves a pair of positions, a tuple (s,t), and the bisimulation relation is defined for a pair of tuples.

Definition 4.7.1 *(Occupation-bisimulation). Two pointed models (M,s,t) and (N,a,b) are said to be occupation-bisimilar, written as $(M,s,t) \Leftrightarrow_B (N,a,b)$, if there exists a relation Z such that $(s,t)Z(a,b)$ for any states $(s,t) \in W^M \times W^M$, $(a,b) \in W^N \times W^N$, whenever $(s,t)Z(a,b)$ the following conditions are satisfied.*

- **Atom 1:** For any atom $p \in P_A$, $s \in V^M(p)$ iff $a \in V^N(p)$
- **Atom 2:** For any atom $p \in P_B$, $t \in V^M(p)$ iff $b \in V^N(p)$
- $Zig\Diamond_A$ - If there exists $u \in W^M$ such that $R^M su$, then there exists $c \in W^N$ such that $R^N ac$ and $(M,u,t)Z(N,c,b)$
- $Zag\Diamond_A$ - If there exists $c \in W^N$ such that $R^N ac$, then there exists $u \in W^M$ such that $R^M su$ and $(M,u,t)Z(N,c,b)$
- $Zig\Diamond_B$ - If there exists $v \in W^M$ such that $R^M tv$, then there exists $d \in W^N$ such that $R^N bd$ and $(M,s,v)Z(N,a,d)$
- $Zag\Diamond_B$ - If there exists $d \in W^N$ such that $R^N bd$, then there exists $v \in W^M$ such that $R^M tv$ and $(M,s,v)Z(N,a,d)$
- $Zig\blacklozenge_A$ - If $\exists v^M, R^M tv, M_v^{p_a}$, then $\exists d \in W^N, R^N bd, N_d^{p_a}$ and $(M_v^{p_a}, s, v)Z(N_d^{p_a}, a, d)$.
- $Zag\blacklozenge_A$ - If $\exists d \in W^N, R^N bd, N_d^{p_a}$, then $\exists v \in W^M, R^M tv, M_v^{p_a}$ and $(M_v^{p_a}, s, v)Z(N_d^{p_a}, a, d)$.
- The corresponding cases for $Zig\blacklozenge_B$, $Zag\blacklozenge_B$ can be defined analogously.

These clauses define the bisimulation relation. Next we will prove the important result that pointed models which are occupation-bisimilar will satisfy the same occupation logic formulas.

Theorem 4.7.2 *For two pointed models (M,s,t) and (N,a,b), if $(M,s,t) \Leftrightarrow_B (N,a,b)$, then (M,s,t) and (N,a,b) are occupation modal equivalent. For occupation modal equivalence refer to 4.2.1.*

Proof. The proof is by induction on the size of formula φ.

Base case 1: $\varphi = p_A, p_A \in P_A$

$(M, s, t) \models p_A \iff s \in V^M(p_A) \iff a \in V^N(p_A)$ (Using **Atom 1** clause of bisimulation)

$a \in V^N(p_A) \iff (N, a, b) \models$ (Using semantics of the logic)

Base case 2: $\varphi = p_B, p_B \in P_B$

$(M, s, t) \models_B$ iff $t \in V^M(p_B)$ iff $b \in V^N(p_B)$ (Using **Atom 2** clause of bisimulation)

$b \in V^N(p_B) \iff (N, a, b) \models p_B$ (Using semantics of the logic)

Induction Hypothesis: The proposition holds for formulas of length L.

Induction Step: To prove that the proposition holds for formulas of length $L + 1$.

Case- *neg*: $\varphi = \neg\psi$

$(M, s, t) \models \varphi \iff (M, s, t) \models \neg\psi \iff (M, s, t) \not\models \psi$

$(M, s, t) \not\models \psi \iff (N, a, b) \not\models \psi$ (Using Induction Hypothesis)

$\iff (N, a, b) \models \neg\psi$

Case $\wedge$: $\varphi = \psi \wedge \chi$

$(M, s, t) \models \psi \wedge \chi \iff (M, s, t) \models \psi \wedge (M, s, t) \models \chi \iff (N, a, b) \models \psi \wedge (N, a, b) \models \chi \iff (N, a, b) \models \psi \wedge \chi \iff (N, a, b) \models \varphi$.

Case- $\Diamond_A$:$\varphi = \Diamond_A\psi$

Let $(M, s, t) \models < left > \psi$, then there exists u, such that $R^M su$, and $(M, u, t) \models \psi$. Since $(M, s, t), (N, a, b)$ are bisimilar, there exists c, such that $R^N ac$, and $(u, t)Z(c, b)$. Then $(N, c, b) \models \psi$(Using Induction hypothesis). Then $(N, a, b) \models \Diamond_A\psi$.

Similarly let $(N, a, b) \models \Diamond_A\psi$, then there exists c such that $R^N ac$ and $(N, c, b) \models \psi$. Since $(M, s, t), (N, a, b)$ are bisimilar, there exists u, such that $R^M su$, such that $(u, t)Z(c, b)$. Then $(M, u, t) \models \psi$ (Using Induction Hypothesis). Then $(M, s, t) \models \Diamond_A\psi$.

Case- $\Diamond_B$: $\varphi = \Diamond_B\psi$ - Proof continues similar to $\Diamond_A$

Case-$\blacklozenge_A$: $\varphi = \blacklozenge_A\psi$

Let $(M, s, t) \models \blacklozenge_A\psi$, then there exists v such that $R^M tv$ and $(M', s, v) \models \psi$, where $M' = (W^M, R^M, v'), V'(p) = V^M(p), V'(p_a) = V^M(p_a) \cup \{v\}$. Now by using **Zig -** $\blacklozenge_A$ and since (M, s, t) and (N, a, b) are bisimilar, there exists d such that $R^N bd$ and $N', a, d) \models \psi$, where $N' = (W^N, R^N, V'), V'(p) = V^N(p), V'(p_a) = V^N(p_a) \cup \{d\}$. Therefore, $(N, a, b) \models \blacklozenge_A\psi$. The other direction can be proven similarly.

Case-$\blacklozenge_B$: $\varphi = \blacklozenge_B\psi$

Let $(M, s, t) \models \blacklozenge_B\psi$, then there exists u such that $R^M su$ and $(M^{p_b}_u, u, t) \models \psi$. Now by using **Zig -** $\blacklozenge_B$ and since (M, s, t) and (N, a, b) are bisimilar, there exists c such that $R^N ac$ and $(N^{p_b}_c, c, b) \models \psi$. Therefore, $(N, a, b) \models \blacklozenge_B\psi$.

The other direction can be proven similarly. This completes the proof. □

Now we are interested in the other direction, given that two models are occupation modally equivalent, are they bisimilar. We know from basic modal logic that this result does not always hold, but holds for image-finite models. We prove a similar result here.

Theorem 4.7.3 *Let M and N be two image finite models, the for every $(s, t) \in W \times W$ and $(a, b) \in N \times N$, if (s, t) and (a, b) are occupation modally equivalent, then $(M, s, t) \Leftrightarrow_B (N, a, b)$.*

Proof. Consider the binary relation E between $M \times M$ and $N \times N$ such that $(s, t)E(a, b)$ if (M, s, t) and (N, a, b) satisfy the same occupation logic formulas. We need to prove that E is the o-bisimulation relation between the two models.
Consider $(s, t) \in M \times M$ and $(a, b) \in N \times N$ such that $(s, t)E(a, b)$. The relation has to satisfy the clauses in o-bisimulation.

Clause : Atom $p \in P_A$: (M, s, t) and (N, a, b) satisfy the same proposition letters $p \in P_A$. This is true because (M, s, t) and (N, a, b) satisfy the same proposition letters by definition of E.

Clause Atom $p \in P_B$: (M, s, t) and (N, a, b) satisfy the same proposition letters $p \in P_B$.This is true because (M, s, t) and (N, a, b) satisfy the same proposition letters by definition of E.
Clause : zig $\Diamond_A$: If there exists $s' \in W^M$ such that $sR^M s'$, then there exists $a' \in W^N$ such that $aR^N a'$ and $(s', t)E(a', b)$.

Assume that there exists no a' such that the above condition is satisfied. Consider the set $T = \{c : aR^N c\}$. T is non-empty otherwise $(N, a, b) \models \Box_A \bot$ while $(M, s, t) \not\models \Box_A \bot$ which is a contradiction. Since we are dealing with image-finite models, T is a finite set such that $T = \{c_1, c_2, ..., c_n\}$. Since no $c \in T$ is such that $(s', t)E(c, b)$, there exists a modal formula ϕ_i, for all $i \in \{1, 2, .., n\}$ such that $(M, s', t) \models \phi_i$ but $(N, c_i, b) \not\models \phi_i$. Therefore, $(M, s, t) \models \diamond(\phi_1 \wedge \phi_2 ... \wedge \phi_n)$ but $(N, a, b) \models \neg(\diamond(\phi_1 \wedge \phi_2 ... \wedge \phi_n))$ which is a contradiction since $(s, t)E(a, b)$. Therefore our assumption is wrong and there exists a a' such that $aR^N a'$ and $(s', t)E(a', b)$. □

Clauses- **zag $\Diamond_A$, zig $\Diamond_B$, zag $\Diamond_B$** Can be proven similar to **zig $\Diamond_A$**

Clause : zig $\blacklozenge_B$ If there exists s' such that $sR^M s'$, $M^{p_b}_{s'}$, then there exists a' such that $aR^N a'$, $N^{p_b}_{a'}$ and $(M^{p_b}_{s'}, s', t)E(N^{p_b}_{a'}, a', b)$

Consider a s' such that $sR^M s'$, $M' = (W_M, R_M, V'_M), V'_M = V_M(p) \forall p \in P, V'_M(p_b) = V_M(p_b) \cup \{s'\}$. Assume that there is no a' such that the above condition is satisfied. Let $T = \{c : aR^N c\}$. T has to be non empty. Since if T is empty, then $(N, a, b) \models \blacksquare_B \bot$ but

$(M, s, t) \nvDash \boxed{B}\bot$. We arrive at a contradiction since they both have to satisfy the same modal formulas. Since we are dealing with image finite models T has to be finite. So, let $T = \{c_1, c_2, .., c_n\}$. Since there is no $c \in T$ such that $(M', s', t)E(N', c, b)$, there exists modal formula $\phi_i \forall i \in [1..n]$ such that $(M', s', t) \models \phi_i$ and $(N', c, t) \nvDash \phi_i$, where $M' = (W_M, R_M, V'_M), V'_M = V_M, V'_M(p_b) = V_M(p_b) \cup \{s'\}$, $N' = (W_N, R_N, V'_N), V'_N = V_N, V'_N(p_b) = V_N(p_b) \cup \{c\}$.
Therefore, $(M, s, t) \models \Diamond_B(\phi_1 \wedge \phi_2 ... \wedge \phi_n)$ but $(N, a, b) \models \neg\Diamond_B(\phi_1 \wedge \phi_2 ... \wedge \phi_n)$, which is a contradiction. There exists an a' such that **zig** $\Diamond_B$ condition is satisfied. $\Box$.
The proofs for the other clauses follows similarly. $\Box$

Occupation modal equivalence does not not always give rise to occupation bisimilarity. The above theorem shows that this is the case for image finite models. Image finite models are special cases of countably saturated models. We will go on to prove that if any two countably saturated models are occupation-equivalent, then they are occupation-bisimilar.

Definition 4.7.4 *(Countably Saturated Models) Let $\mathfrak{M} = (W, R, V)$ be a model of the basic modal logic, X be a subset of W and Σ a set of modal formulas. Σ is satisfiable in the set X if there is a state $x \in X$ such that $M, x \models \varphi$ for all $\varphi \in \Sigma$. Σ is finitely satisfiable in X if every finite subset of Σ is satisfiable in X.*
The model $\mathfrak{M}$ is called countably saturated if it satisfied the following conditions for every state $w \in W$ and every set Σ of modal formulas.

If Σ is finitely satisfiable in the set of successors of w,
then Σ is satisfiable in the set of successors of w

Looking at it from a first order perspective, considering the model $M = (W, R, V)$(viewing as a FOL structure). A set of FOL formulas $\Gamma(x)$ from $\mathfrak{L}$ with one free variable x is realized by M if there exists $w \in W$ such that $M \models \tau(x)[x := w]$ for all $\tau(x) \in \Gamma(x)$. We say that M is countably saturated if for every finite set $X \subset W$, the expansion M_X realizes every set $\Gamma(x)$ in L_X(the expansion of $\mathfrak{L}$ with constants for the elements in X) whenever every finite subset $\Gamma'(x) \subset \Gamma(x)$ is satisfied in M_X.

Theorem 4.7.5 *Let M and N be two countably saturated models. If (M, s, t) and (N, u, v) satisfy the same occupation logic formulas they are occupation-bisimilar and the bisimilarity relation is given by modal equivalence.*

Proof. Consider the two countably saturated models (M, s, t) and (N, u, v) that satisfy the same occupation logic formulas. We define a relation $E = \{((s, t), (u, v)) | (s, t) \in M_w \times M_w, (u, v) \in N_w \times N_w, (M, s, t)$ and (N, u, v) satisfy the same occupation logic formulas $\}$.
Now we move on to prove that the relation E defined above satisfy the clauses of occupation bisimulation.

Case : Atom 1: $p \in P_A$

Since (M, s, t) and (N, u, v) satisfy the same occupation logic formulas, this case holds trivially.

Case : Atom 2: $p \in P_B$

Since (M, s, t) and (N, u, v) satisfy the same occupation logic formulas, this case holds trivially.

Case : $zig\Diamond_A$

Suppose there exists $s' \in M_w$ such that $R^M ss'$, we need to show that there exists $u' \in N_w$ such that $R^N uu'$ and $(M, s', t)E(N, u', v)$.
Let Σ be the set of formulas true at (s', t). For every finite subset Δ of Σ, $(M, s', t) \models (\wedge\Delta)$, hence $(M, s, t) \models \Diamond_A \wedge \Delta$. As (s, t) is logically equivalent to (u, v), $(N, u, v) \models \wedge\Delta$. So there exists a successor u_Δ to u such that $(N, u_\Delta, v) \models (\wedge\Delta)$. Σ is finitely satisfiable in the set of successors of u.
Therefore, by countably saturation Σ itself is satisfiable is (u', v).So $(s', t)E(u', v)$
Proof for **Cases:** $zag\Diamond_A, zig\Diamond_B, zag\Diamond_B$ proceed similar to $zig\Diamond_A$

Case : $zig\blacklozenge_B$

There exists $R^M ss'$ and $M^{p_b}_{s'}$. We need to show that there exists $R^N uu'$ and $(M^{p_b}_{s'}, s', t)E(N^{p_b}_{u'}, u', v)$.
Let Γ be the set of formulas true at $(M^{p_b}_{s'}, s', t)$. Let Δ be a finite subset of Γ. The following inferences hold. By theorem 4.6.3
$(M, s, t) \models \blacklozenge_B \wedge \Delta \iff (N, u, v) \models \blacklozenge_B \wedge \Delta$
$\iff N \models ST^{\emptyset,\emptyset}_{x,y}(\blacklozenge_B \wedge \Delta)[s, t]$
$\iff N \models z(xRz \wedge ST^{\{z\},\emptyset}_{z,y}(\wedge\Delta))[s, t]$
Since N is countably saturated,
$\exists z \in N_w, N \models ST^{\{z\},\emptyset}_{z,y}(\Gamma)$.
Therefore, there exists a pointed model $N^{p_b}_{u'}$ such that $R^N uu'$ and $N \models ST^{\emptyset,\emptyset}_{x,y}(\Gamma)[u', v]$.

Therefore using Theorem 4.6.3 $(s', t), (u', v)$ are occupation modally equivalent and hence bisimilar by our definition of the bisimulation relation.
Proof for **Cases:** $zag\blacklozenge_B, zig\blacklozenge_A, zag\blacklozenge_A$ proceed similar to $zig\blacklozenge_B$.
This completes the proof. □

4.8 Characterization of FO Formulas

In this section we characterize the first order logic (FOL) formulas which are equivalent to translations of occupation logic formulas. The van Benthem Characterisation theorem states that for basic modal logic, only those FOL formulas which are invariant under bisimulation are logically equivalent to translations of basic modal logic formulas. Similarly, we will prove that FOL formulas that are invariant under the occupation bisimulation are exactly those formulas that are logically equivalent to standard translation of occupation logic formula.

A first order formula $\varphi(x, y)$ is invariant under occupation bisimulation if for all models

M and N and all pairs of states (s,t) in M and (u,v) in N, and all bisimulations E between M and N such that $(s,t)E(u,v)$, $M \models \varphi(x,y)[s,t]$ iff $N \models \varphi(x,y)[u,v]$.

Theorem 4.8.1 *Let $\varphi(x,y)$ be a first order formula. Then $\varphi(x,y)$ is invariant under occupation bisimulation iff $\varphi(x,y)$ is logically equivalent to standard translation of an occupation logic formula.*

Proof. **If-part:**Let $\varphi(x,y)$ be a first order formula and $\varphi(x,y)$ is logically equivalent to a standard translation of an occupation logic formula. To show that it is invariant under occupation bisimulation.
Let $\varphi(x,y)$ be equivalent to standard translation $ST^{\emptyset,\emptyset}_{x,y}(\alpha)$ of a basic occupation logic formula α,say.
Take two models M,N and states (s,t) in M and (u,v) in N such that (M,s,t) is occupation bissimilar to (N,u,v).Then $(M,s,t) \models \alpha$ iff $(N,u,v) \models \alpha$.
Thus $M \models \varphi(x,y)[s,t] \iff M \models ST^{\emptyset,\emptyset}_{x,y}(\alpha)[s,t] \iff (M,s,t) \models \alpha \iff (N,u,v) \models \alpha \iff N \models ST^{\emptyset,\emptyset}_{x,y}(\alpha)[u,v] \iff N \models \varphi(x,y)[u,v]$.
The proof is completed.
Only-If Part Let $\varphi(x,y)$ be a first order formula which is invariant under occupation bisimulations. It is to be shown that $\varphi(x,y)$ is logically equivalent to standard translation of a basic occupation logic formula.
ModCon(φ) = $\{ST^{\emptyset,\emptyset}_{x,y}(\alpha) |\ \alpha$ is a occupation formula and $\varphi(x,y) \models ST^{\emptyset,\emptyset}_{x,y}(\alpha)\}$
Claim: If $ModCon(\varphi) \models \varphi(x,y)$, then $varphi(x,y)$ is logically equivalent to standard translation of a basic occupation formula.

Proof of Claim : Suppose $ModCon(\varphi) \models \varphi(x,y)$, then by compactness of FOL, there is a finite subset χ of $ModCon(\varphi)$ such that $\chi \models \varphi(x,y)$. So $\wedge\chi \models \varphi(x,y)$ which implies $\models \wedge\chi \to \varphi(x,y)$. And we also have $\models \varphi(x,y) \to \wedge\chi$.So $\models \varphi(x,y) \iff \wedge\chi$. Now every β in χ is a occupation logic formula and so is $\wedge\chi$. Therefore $\varphi(x,y)$ is logically equivalent to standard translation of a occupation formula. □

To show $ModCon(\varphi) \models \varphi(x,y)$ we need to show that for all models M and for all pairs of states (s,t), if $M \models ModCon(\varphi)[s,t]$, then $M \models \varphi(x,y)[s,t]$.

Proof Let $G(x,y) = \{ST^{\emptyset,\emptyset}_{x,y}(\alpha) |\ \alpha$ is an occupation formula and $M \models ST^{\emptyset,\emptyset}_{x,y}(\alpha)[s,t]$.
We need to show that $G(x,y) \cup \varphi(x,y)$ is consistent. Suppose not, then by compactness of FOL, there is a finite subset $H(x,y)$ of $G(x,y)$ such that $\models \wedge H(x,y) \to \neg\varphi(x,y)$ that is $\models \varphi(x,y) \to \wedge H(x,y)$. So $\neg \wedge H(x,y) \in ModCon(\varphi)$. But then $M \models \neg H(x,y)[s,t]$. We arrive at a contradiction. Therefore $G(x,y) \cup \varphi(x,y)$ is satisfiable.
Consider a pointed model (N,u,v) such that $N \models G(x,y) \cup \varphi(x,y)[u,v]$. It can now also be said that (M,s,t) and (N,u,v) are occupational modal equivalent.
Consider two countably saturated elementary extensions (M_ω,s,t) and (N_ω,u,v) of (M,s,t) and (N,u,v) respectively. As FOL is invariant under elementary extensions, from $N \models \varphi(x,y)[u,v]$ we can conclude that $N_\omega \models \varphi(x,y)[u,v]$. As we have assumed that φ is invariant

to occupation bisimulation and using Theorem 4.8 we can say that $M_\omega \models \varphi(x,y)[s,t]$ which gives us $M \models \varphi(x,y)[u,v]$. □

The proof concludes here. □

Chapter 5

Occupation Logic: More Properties

5.1 Tree Model Property

Basic modal logic has the tree model property. For any model M, we can construct a bisimilar model M' which has a tree-like structure. Evidently, if any logic has ability to show the existence of cycle in the frame (W, R) then the logic does not enjoy the tree-model property. Consider the following fact for occupation logic,

Fact 5.1.1 *Consider the class of formulas δ_n with $n \in \mathbb{N}_{>0}$ defined inductively as follows: $\delta_1 = \Diamond p_a$; $\delta_{i+1} = \Diamond(\neg p_a \wedge \delta_i)$.*
Let $M = (W, R, V)$ be a model such that $V(p_a) = V(p_b) = \emptyset$. Then, there exists $w \in W$ such that $(M, w, w) \models \blacklozenge \Diamond \delta_n$ if and only if there exists a circuit of length n in the frame (W, R).

Proof. The formula $\blacklozenge \Diamond \delta_n$ has only one occurrence of $\blacklozenge$ modality. As we have $V(p_a) = \emptyset$ initially, the only poisoned state when we go through the formula is the one poisoned by $\blacklozenge$. The formula says that it is possible to go through n non-poisoned states before reaching the poisoned state. So we can say there exists a cycle of length $\leq n$. □

As a consequence of the above fact, Occupation Logic does not have the tree model property.

5.2 Finite Model Property

A logic has finite model property if given any formula φ, there exists a finite model (a model with finite number of states) which satisfies the formula. To show the absence of finite model property, it is enough to come up with a formula which can only be satisfied by infinite models(models having infinite number of states). Here we show that occupation logic lacks the finite model property.

Proposition 5.2.1 *Occupation Logic does not have the finite model property.*

Proof. We provide a formula that can only be satisfied in an infinite model. Let us consider $\varphi = \alpha \wedge \beta \wedge \gamma \wedge \delta \wedge \eta$ with the formulas defined below. The idea of using these formulas will be explained later.

- $\alpha = \neg q \wedge \Diamond \wedge \square q \wedge \square(\Diamond \wedge \square \neg q)$: The current state falsifies q and all its successors ($\diamond$ ensures there is atleast one) satisfy q and have in turn successors (at least one) which all falsify q.

- $\beta = \blacksquare \square \Diamond p_a$: after poisoning a state is reached whose successors can reach the poisoned state in one step, i.e., all successors of the current state have successors linked via symmetric edges.

- $\gamma = \blacksquare \square \square \Diamond(\neg q \wedge \Diamond p_a) \wedge \square \square \neg \blacklozenge \Diamond p_a$: after any poisoning a state is reached whose successors are not reflexive, the right conjunct ensures that, and can reach a $\neg q$ state which can in turn reach the poisoned state. In simpler terms, all successors of the current state lay on cycles of length 3.

- $\delta = \square \square \blacksquare \square \square (q \rightarrow \Diamond p_a)$ all successors of the current state's successors are such that after any poisoning, further q-successors can reach back to the poisoned state.

- $\eta = \square \blacklozenge \Diamond \neg \Diamond (q \wedge \Diamond(\neg q \wedge \Diamond p_a))$: all successors of the current state are such that there is one successor that can be poisoned and such that none of its successors satisfies q and can reach the poisoned state in two steps via a $\neg q$ state.

Now, let $(M, w, w) \models \varphi$. Then w is followed by distinct successors $w'(\alpha)$ that have successors w'' which are linked back to their predecessors w' by symmetric edges (implied by β). These w'' states also have successors, different from w' which also have w' as successor(γ) and which are also successors of $w'(\delta)$. Hence w'' is followed by an infinite path of distinct states. Finally there exists one such w'' which has no other predecessor than $w'(\eta)$, that is w'' is the root of an infinite sequence of distinct states which are all successors of w'. □

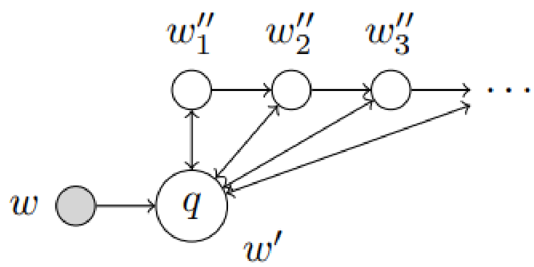

Figure 5.1: A model that satisfies φ

5.3 Decidability

In this section we establish the undecidability of Occupation Logic by reduction from Tiling problem 3.5.4.

Given: An OL formula φ

Problem: Is there a pointed model (M, s, t) with $V(p_a) = V(p_b) = \emptyset$ such that $(M, s, t) \models \varphi$.

Theorem 5.3.1 *The satisfaction problem for OL is undecidable.*

Proof. We reduce the problem of $\mathbb{N} \times \mathbb{N}$ tiling problem to the satisfaction problem. The tiling problem is described in 3.5.4.

Let T be a finite set of tiles. For a tile $t \in T$, the predicate p_t models the fact that t is placed on the point and the four predicates $top(t), right(t), left(t), bottom(t)$ represent the four colour of t.

We claim that,

φ is satisfiable if and only if T tiles the grid $\mathbb{N} \times \mathbb{N}$.

To do the proof we will use three relations in the model $M = (W, R, R_1, R_2, V)$. Now we have three modalities for A's moves $\langle A\rangle, \langle A\rangle_1, \langle A\rangle_2$ and three poison modalities $\blacklozenge_B, \blacklozenge_{B,1}, \blacklozenge_{B,2}$. Similarly for B's moves we extend with $\langle B\rangle, \langle B\rangle_1, \langle B\rangle_2$ and three poison modalities $\blacklozenge_A, \blacklozenge_{A,1}, \blacklozenge_{A,2}$. These relations can be reduced to one relation as in our standard model by making use of arguments in [KW99].

In these formulas, the modality $\langle A\rangle_1, \langle B\rangle_1$ represent vertical moves on the grid, the modalities $\langle A\rangle_2, \langle B\rangle_2$ represent horizontal moves and the modality $\langle A\rangle, \langle B\rangle$ moves from any point to any point on the grid.

Let $\varphi_T = \alpha \wedge \beta \wedge \gamma \wedge [A](\delta_T^1 \wedge \delta_T^2 \wedge \delta_T^3)$ with $\alpha, \beta, \gamma, \delta_T^1, \delta_t^2, \delta_T^3$ defined below. For the purpose of the proof we consider $p_t \in P_A \forall t \in T$ and $qinP_A$. The formulas are interpreted in the pointed model (M, w, w).

- $\alpha = q \wedge [A](\neg q \wedge \langle A\rangle q) \wedge [A][B]_1[A]_1\langle A\rangle(q \wedge \langle A\rangle p_a) \wedge [A][B]_2[A]_2\langle A\rangle(q \wedge \langle A\rangle p_a)$
  w satisfies q and its R-successors do not satisfy q and have an edge back to w and the set of its R-successors is closed under R_1 and R_2. This formula establishes w as a spy point which links to all the points on the $\mathbb{N} \times \mathbb{N}$ grid which also link back to the spy point. It also adds that the set of successors of any grid point reached using the R_1 and R_2 is always reachable from w using the R relation.

- $\beta = \bigwedge_{i=1,2}([A]\langle A\rangle_i \wedge [B][A][A](q \rightarrow [A](\langle A\rangle_i p_a \rightarrow [A]_i p_a)))$
  For all successors of w using the R-relation, the relations R_1 and R_2 are total functions. All the points in the grid have one successor to the right and one successor above.

- $\gamma = \blacksquare_A[A][A](q \rightarrow [A]([A]_1[A]_2\neg p_a \vee [A]_2[A]_1 p_a))$
  The relations R_1 and R_2 commute. This formula establishes the grid like structure.

In a grid the sequence of steps moving to the right and moving above can be done in any order and we will reach the same grid point.

- $\delta_T^1 = \bigvee_{t\in T}(p_t \wedge \bigwedge_{t'\in T, t'\neq t} \neg p_{t'}))$
  At each grid point, we place only one tile t from the finite set of tiles T.

- $\delta_T^2 = \bigwedge_{t\in T}(p_t \rightarrow \boxed{A}_2 \bigvee_{t'\in T, left(t')=right(t)} p_t)$
  This formula establishes that any two tiles present vertically, have the same colour on the common edge.

- $\delta_T^3 = \bigwedge_{t\in T}(p_t \rightarrow \boxed{A}_1 \bigvee_{t'\in T, bottom(t')=top(t)} p_t)$
  This formula establishes that any two tiles present horizontally, have the same colour on the common edge.

- $\boxed{A}(\delta_T^1 \wedge \delta_T^2 \wedge \delta_T^3)$
  Only one tile is present at each node and horizontal and vertical tiling are correct

Based on the analysis of the formulas which comprise φ_T, all models that satisfy φ_T is a tiling of $\mathbb{N} \times \mathbb{N}$
We need to show the other direction of the equivalence. Suppose there is a function $f : \mathbb{N} \times \mathbb{N} \rightarrow T$ is a tiling of $\mathbb{N} \times \mathbb{N}$. Define a model $M_T = (W, R, R_1, R_2, V)$ in the following way.

- $W := \{w\} \cup \mathbb{N} \times \mathbb{N}$
- $R =$ For all $x \in \mathbb{N} \times \mathbb{N}$, $(w, x), (x, w) \in R$
- $R_1 = \{((n, m), (n+1, m)) | n, m \in \mathbb{N}\}$
- $R_2 = \{((n, m), (n, m+1)) | n, m \in \mathbb{N}\}$
- $V(p_t) = \{(n, m) \in \mathbb{N} \times \mathbb{N} | f(n, m) = t\}$ for all $t \in T$
- $V(q) = \{w\}$.

By construction of the model M_T, we can check that $M_T \models \varphi_T$. □

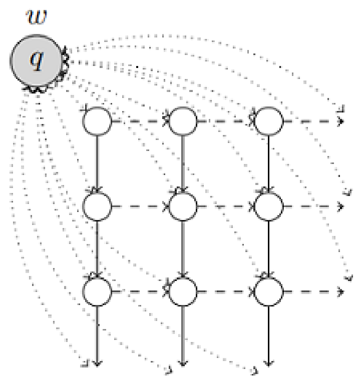

Figure 5.2: A model of formula φ_T, R is represented by dotted links, R_1 by dashed links and R_2 by plain links. The shadowed state is the spy point w

5.4 Model Checking

Model Checking refers to the problem of given a formula φ and a model M whether $M \models \varphi$. Let φ be an occupation logic formula. Consider the model $M = (W, R, V)$. Let $s, t \in W$. Any model checking algorithm attempts to answer whether $(M, s, t) \models \varphi$. We give a recursive algorithm for model checking, prove its correctness. We also give a upper bound on the time complexity of model checking.

5.4.1 Algorithm for Model Checking

The algorithm 1 is a recursive algorithm that takes as input, the Model M, the states s, t where the evaluation has to be performed and the formula φ. Along with this, since poisoning of states occur during the evaluation of the formula, we keep track of set of states poisoned for A and B using the sets $poisonA, poisonB$, respectively. Initially $poisonA = poisonB = \emptyset$. The algorithm solves the original problem by recursively considering the subformulas. The proof of correctness of algorithm follows from the algorithm itself since it is a recursive algorithm.

5.4.2 Upper Bound of Model Checking

Theorem 5.4.1 *The upper bound for model checking in Occupation Logic is given by* $O(|\varphi|.|M|^3.2^{|M|})$

Proof. Given a pointed model M, s, t and a formula φ the upper bound for model checking can be established using algorithm1. The recursive algorithm proceeds by finding the answer of whether $M, u, v \models \psi$ for every sub-formula ψ of φ , for all states $u, v \in W$ and every possible subset of poisoned states $poisonA, poisonB$.

The answers for such sub-problems are stored (memorized). The algorithm 1 does not

memorize the answers but it can be modified to store the answers by following same techniques used for recursive algorithms as mentioned in [CLRS03]. The time complexity is contributed by the number of sub-problems involved and the time taken to solve each sub-problem.
Any sub-problem involves $(u, v, \psi, poisonA, poisonB)$. We calculate the number of sub-problems by looking at each contributing factor individually.

- The factor of $O(|M|^2)$ occurs because, we need to consider pair of states u, v for evaluating any subformula.
- The factor of $O(2^{|M|})$ occurs because, whenever we consider any sub-problem $M, u, v \models \psi$, we also need to know the set of states poisoned till this step in the algorithm for subsequent evaluation. The number of possible subsets of M is $2^{|M|}$ which contributes to the complexity.

- the factor $O(|\varphi|)$ occurs since we are considering every sub-problem.

Now solving for time complexity of each sub-problem.

- For formulas of the form $< A > \psi$, where $< A >= \blacklozenge_A, \blacklozenge_B, \blacksquare_A, \blacksquare_B, \boxed{A}, \boxed{B}, \Diamond_A, \Diamond_B$, we need to visit the neighbours of the current state which take linear time, $O(M)$ if we use any standard graph traversal algorithm.
- For formulas of the form $\psi \wedge \chi, \psi \vee \chi, \neg\psi$, it takes constant time.
- The propositional variables form the base cases and solving them involves testing for set membership which can be performed in constant time.

Therefore, the upper bound on time complexity is given by $O(|\varphi|.|M|^3.2^{|M|})$. $\square$

Input: Pointed Model M, s, t, formula φ, Sets $poisonA = \emptyset, poisonB = \emptyset$
Output: $True$ if formula is satisfiable, otherwise $False$
Function `Check`(φ,M, s, t,$poisonA$,$poisonB$)**:**

```
if φ = p_a then
    return s ∈ V(p_a) ∨ s ∈ poisonA
if φ = p_b such that p ∈ P_A then
    return t ∈ V(p_b) ∨ t ∈ poisonB
if φ = p such that p ∈ P_A then
    return s ∈ V(p)
if φ = p such that p ∈ P_B then
    return t ∈ V(p)
if φ = (ψ ∧ χ) then
    return
     (Check(ψ, M, s, t, poisonA, poisonB) ∧ Check(χ, M, s, t, poisonA, poisonB))
if φ = (ψ ∨ χ) then
    return
     (Check(ψ, M, s, t, poisonA, poisonB) ∨ Check(χ, M, s, t, poisonA, poisonB))
if φ = ¬ψ then
    return ¬Check(ψ, M, s, t, poisonA, poisonB)
if φ = [A]ψ then
    for each successor r of s do
        if Check(ψ, M, r, t, poisonA, poisonB) == False then
            return false
    return True
if φ = ⟨A⟩ψ then
    for each successor r of s do
        if Check(ψ, M, r, t, poisonA, poisonB) == True then
            return True
    return False
if φ = ■Bψ then
    for each successor r of s do
        if Check(ψ, M, r, t, poisonA ∪ {r}, poisonB) == False then
            return False
    return True
if φ = ◆Bψ then
    for each successor r of s do
        if Check(ψ, M, r, t, poisonA ∪ {r}, poisonB) == True then
            return True
    return False
/* The logic for [B]φ, ⟨B⟩φ, ■Aφ, ◆Aφ follow similarly                    */
return
```

Algorithm 1: Model Checking Problem

Chapter 6

CONCLUSION

In this work, the initial goal was to develop a logic, for discussing about the Occupation Game. Towards the goal, we have developed occupation logic and study its logical properties in depth. We give a brief summary of the topics discussed and give scope and directions for future work.

6.1 Summary

Chapter 1 gives a introductory view of games in general, how their mathematical analysis is beneficial and introduces the topic of study of the thesis, the occupation game. This work is expanded upon in Chapter 2 which discusses poison game, a simplified version of occupation game and compares the similarities and differences between the two. The terminology that will be used for further discussion is also introduced.

A brief discussion of modal logic follows in Chapter 3. Since occupation logic is an extension of modal logic, the chapter serves as an overview of the properties of modal logic that will be investigated for the new logic introduced. Chapter 4 and Chapter 5 studies Occupation Logic in depth. The following topics are discussed, among others.

- Occupation Bisimulation
- Finite Model Property(FMP)
- Decidability
- Model Checking

6.2 Future Work

The logic introduced is very expressive and is undecidable. An advantage of reducing the expressivity of the logic is that we might obtain a logic that is decidable, and thus may be of practical use. Such a logic can also allow us to find certain decidable fragments of FOL. Another direction of study can be in establishing tighter lower bounds for model checking.

Bibliography

[AF84] M Aigner and M Fromme. A game of cops and robbers. *Discrete Applied Mathematics*, 8:1–12, 1984.

[AvBG18] Guillaume Aucher, Johan van Benthem, and Davide Grossi. Modal logics of sabotage revisited. *Journal of Logic and Computation, Oxford University Press(OUP)*, 28(2):269–303, 2018.

[BdRV10] Patrick Blackburn, Maarten de Rijke, and Yde Venema. *Modal Logic*. Cambridge Tracts in Theoretical Computer Science, 2010.

[CLRS03] Thomas H. Cormen, Charles E. Leriserson, Ronald L. Rives, and Clifford Stein. *Introduction to Algorithm*. The MIT Press, 2003.

[DM93] P Duchet and H Meyniel. Kernels in directed graphs: a poison game. *Discrete Mathematics*, 115:273–276, 1993.

[Gar20] James Garson. Modal logic. In *The Stanford Encyclopedia of Philosophy(Summer 2021 Edition),Edward N. Zalta(ed.)*, June 2020.

[GR19] Davide Grossi and Simon Rey. Credulous acceptability, poison games and modal logic. In *AAMAS'19: Proceedings on the 18th conference on Autonomous Agents and MultiAgent Systems*, May 2019.

[Gru64] P.M Grundy. Mathematics and games, 1964.

[KW99] M. Kracht and F. Wolter. Normal monomodal logic can simulate all others. *Journal of Symbolic Logic*, 64:99–138, 1999.

[Mye91] Roger B. Myerson. *Game Theory: Analysis of Conflict*. Harvard University Press, 1991.

[Nas49] John F. Nash. Equilibrium points in n-person games. *Proceedings of the national academy of the sciences of the USA*, 36(1):48–49, 1949.

[Rob71] Raphael M. Robinson. Undecidability and nonperiodicity for tilings of the plane. *Inventiones mathematicae*, 12:177–209, 1971.

[Sch02] Philippe Schnoebelen. The complexity of temporal logic model checking. In *Advances in Modal Logic 4, papers from the fourth conference on "Advances in Modal Logic"*. King's College Publications, 2002.

[vBK20] Johan van Benthem and Dominik Klein. Logic for analyzing games. In *The Stanford Encyclopedia of Philosophy(Summer 2020 Edition),Edward N. Zalta(ed.)*, June 2020.

[vBL20] Johan van Benthem and Fenrong Liu. Graph games and logic design. In *Knowledge, Proofs and Dynamics*. Springer, March 2020.