Some Decidable Classes of the Distributed Synthesis Problem

By

Anup Basil Mathew

MATH10201005004

The Institute of Mathematical Sciences, Chennai

A thesis submitted to the

Board of Studies in Mathematical Sciences.

In partial fulfillment of requirements

for the Degree of

DOCTOR OF PHILOSOPHY

of

HOMI BHABHA NATIONAL INSTITUTE

December, 2017

Homi Bhabha National Institute

Recommendations of the Viva Voce Board

As members of the Viva Voce Board, we certify that we have read the dissertation prepared by Anup Basil Mathew entitled "Some Decidable Classes of the Distributed Synthesis Problem" and recommend that it maybe accepted as fulfilling the dissertation requirement for the Degree of Doctor of Philosophy.

____ Date:

Chairman -

____ Date:

Guide/Convener -

____ Date:

Examiner -

____ Date:

Member 2 -

____ Date:

Member 3 -

Final approval and acceptance of this dissertation is contingent upon the candidate's submission of the final copies of the dissertation to HBNI.

I hereby certify that I have read this dissertation prepared under my direction and recommend that it may be accepted as fulfilling the dissertation requirement.

Date:

Place: Guide

STATEMENT BY AUTHOR

Anup Basil Mathew

DECLARATION

I, hereby declare that the investigation presented in the thesis has been carried out by me. The work is original and has not been submitted earlier as a whole or in part for a degree / diploma at this or any other Institution / University.

Anup Basil Mathew

List of Publications arising from the thesis

Journal :

1. ”Infinite games with Finite Knowledge Gaps”, Dietmar Berwanger and Anup Basil Mathew, *Information and Computation*, **2017**, Vol. 254, pp. 217-237.

Conferences :

1. ”Games with Recurring Certainty”, Dietmar Berwanger and Anup Basil Mathew, *Proceedings of the 2nd International Workshop on Strategic Reasoning*, **2014**, Vol. 146 of EPTCS, pp. 91-96.
2. ”Hierarchical Information Patterns and Distributed Strategy Synthesis”, Dietmar Berwanger, Anup Basil Mathew and Marie van den Bogaard, *Proceedings of the 13th International Symposium on Automated Technology for Verification and Analysis*, **2015**, Vol. 9364 of LNCS, pp. 378-393.

Anup Basil Mathew

ACKNOWLEDGEMENTS

I thank my guide R.Ramanujam for his guidance and support. His advice in both academic and other matters have been of great value to me. His constant encouragement and willingness to accommodate my quirks has been a big factor in my steady improvement as a student. Most importantly, he allowed me to grow as a researcher at a pace that suited me, and for this I am extremely grateful.

I thank Dietmar Berwanger for inviting me to LSV, during which time I learnt many interesting aspects of the problem studied here. His willingness to listen to and correct my poorly formed ideas have been crucial to my understanding of the subject. I also thank him for being extremely generous in making my stay at LSV comfortable.

I thank the Director V. Arvind for allowing an extension of my tenure here at IMSc, and I also thank Vijay Kodiyalam for his help with the thesis submission process.

Contents

List of Figures

Synopsis

The synthesis problem aims to find an algorithm, that when given a specification, generates a program whose behaviour satisfies the given specification. When the synthesized program is required to be a *distributed program*, that is, as a collection of interacting programs implemented on a distributed system, we call it the *distributed synthesis problem*.

In this thesis, we study the distributed synthesis problem for certain simple distributed systems that have the following operational template: The distributed system operates in rounds. Each sub-system of the distributed system has access to read or write on some boolean variables. At each round, the environment writes on some of the variables and each program implemented on a sub-systems first reads the variables it has access to and then writes on the variables it has access to. The *behaviour* of such a distributed system is given by the sequence of reads and writes performed in each round. By a specification of a distributed program we mean a property of the behaviour of the distributed program.

The distributed synthesis problem admits a natural correspondence with the *winning strategy problem* for *multi-agent imperfect information games* and so we study this formulation of the distributed synthesis problem. The correspondence can be explained informally as follows: The input specification for the distributed synthesis problem is transformed to an adversarial game between a *coalition* of agents and an environment-agent. The distributed system now becomes a 'action-labelled game graph', where the vertices of the game graph represent the states of the system, and the edges represent the state change due to the read/write actions that label the edge. The behavioural specification becomes the winning condition for the coalition. The game tree for this game is given by an ‘unravelling’ of this game graph, where the nodes of the tree represent partial runs of the distributed system and the ‘indistinguishability relation’ for an agent relates partial runs that are informationally identical to an agent. A winning strategy for the coalition now corresponds to a distributed program that satisfies the desired specification.

It is known due to [1] that the winning strategy problem for multi-agent imperfect information games is unsolvable, even for very simple games and winning conditions. In the equivalent setting of the distributed synthesis problem, it is known due to [2] that the synthesis problem is unsolvable for distributed systems specified by means of certain simple *architectures* with ‘LTL specifications’. They go on to show that the synthesis problem is solvable for ‘hierarchical architectures’ with such specifications. Later it was shown in [3] that the solvability can be extended to a larger class of architectures and for global μ-calculus specifications, using the narrowing construction that they had used earlier. The state-of-the-art result in the case was given in [4], where a criterion for architectures called the ‘information forks’ is introduced, the absence of which is a necessary and sufficient condition for solving the distributed synthesis problem for μ-calculus specifications. The study of ‘local specifications’ for the synthesis problem was introduced in [5]. A local specification is essentially a weaker notion of specification which is given as a conjunction of the specifications for each sub-system of the distributed system. In this work they give a necessary and sufficient condition for solving the distributed synthesis problem on ‘acyclic architectures’ with local specifications given in linear temporal logics. Later in [6] that this solution can be extended to the case of ‘flanked-pipelines with feedback’. We remark here that this list of results is not meant to be comprehensive, but rather illustrative of the state of research on this problem.

We study the winning strategy problem for multi-agent imperfect information games with μ-calculus winning conditions (both local and otherwise), with an eye towards the distributed synthesis problem. We remark here that the specification language of μ-calculus is at least as expressive as all the specification languages considered above. Our goal is to obtain a purely 'game-based' solutions to the winning strategy problem, by means other than the automata-theoretic solutions, which is the conventionally followed approach. The advantage of the game approach is the robustness it offers to slight changes in the problem.

The contributions of this thesis can be broadly divided into four parts:

- We study a class of imperfect information games called *games with recurring common knowledge of state*, and show that such games are solvable for winning conditions given by 'priority labelling' of the game graph. This is intended as a departure from the class of hierarchical games, which are the only previously known case of multi-agent imperfect information games that admit a solution to the winning strategy problem. We show that if a game satisfies the property of recurring common knowledge of state, then there is an NEXPTIME algorithm that decides whether a winning strategy exists for the game. Moreover, the winning strategy can be synthesized in 2EXPTIME.
- We study a generic approach to solving imperfect information games, called the *retraction approach*. The key idea here is the design of a class of structures, called 'witnesses', that witness the existence of a winning strategy in the game. Then we define a generic operation called 'retraction', that transforms witnesses to possibly smaller witnesses. We show that if a class of witnesses of a game, called *canonical witnesses*, admit retraction to small witnesses, then it is a sufficient criteria for solving the winning strategy problem.
- We consider the case of of architectures that admit *weak-informedness ordering* and *informedness ordering* on its agents. These classes are known to have a decidable distributed synthesis problem, and we give new proofs for these via the retraction approach. We show that these architecture classes correspond to games that contain no *uniform-determinsitic fork-triples* and *fork-triples* respectively, and that canonical witnesses obtained from such games belong to a class of witnesses called *modular*

witnesses. To solve the corresponding winning strategy problem, we show that modular witnesses admit retractions to small witnesses.

- We introduce the class of *weak-broadcast architectures* and *broadcast architectures*, and show that for the case of local specifications, these architecture have a decidable distributed synthesis problem. The solution for this proceeds in two steps: first we show that canonical witnesses for games corresponding to these classes can be *factorized* into simpler witnesses, and second, we show that such witnesses admit retractions to small witnesses by means of the retractions of their 'factors'. We note that these architecture classes subsume the known decidable classes in the case of local specifications.

Chapter 1

Introduction

A central challenge in computer science is to find algorithms that can determine if a program meets a desired specification. This challenge has been approached in two different ways, as a *verification problem* and a *synthesis problem*.

The verification problem aims to find algorithms, that when given a program and a specification as input, determines if the behaviour of the program satisfies the given specification. A fundamental result in this direction is Rice's theorem [7], which says that any 'behavioural' property of a program, for example the property of termination of a program, cannot be determined procedurally. Therefore the focus shifts to solving the verification problem for classes of programs, that have either a finite state-space (that is, a finite-state automaton) or an infinite but structured state-space (for example, a push-down automaton).

On the other hand the synthesis problem aims to find an algorithm, that when given a specification, generates a program whose behaviour satisfies the given specification, when it exists. Clearly, this is a harder problem as compared to the verification approach, but with the obvious advantage that one now obtains a program that satisfies the specification. Here again, the focus shifts to synthesizing programs that have either a finite state-space, or an infinite but structured state-space.

The subject matter of this thesis pertains to the synthesis problem. Among the many variants of this question, depending on the chosen model of computation and the specification language, our attention here is on models of computation that are *reactive distributed systems*, and the specification language of μ*-calculus*. We call the synthesis problem in this setting, as the *distributed synthesis problem*. We now describe the terminology referenced here.

The standard notion of computation is that of a Turing machine, which operates as follows: an input is presented to the Turing machine, which is then processed, and an output delivered. However, many systems like operating systems, hardware controllers, etc. which also perform computation, act quite differently. Such systems interact for ever with an environment, and perform actions to maintain some invariant property of the system. The term *reactive* refers to such systems, that interact continually with an environment. Additionally, such systems may themselves be a collections of many interacting sub-systems, in which each sub-system receives only a part of the input from the environment. The term *distributed* refers to such systems, and we use the term *centralized* to refer to systems that are not composed of multiple sub-systems. Moreover, we assume that the system is *synchronous*, that is, it has a global clock that is available to all sub-systems. We note that the distributed synthesis question without the assumption of the system being synchronous has also been studied, for example in [8], [9], but here we focus only on the synchronous case.

The notions of program and specification for reactive distributed systems differs significantly from that of Turing machines and reactive centralized systems. To distinguish this notion of a program, we call it a *distributed program*. We list below some of its characteristics.

- Due to the system being distributed, one must now synthesize a distributed program, which is a collection of *sub-programs*, one for each sub-system. At each instant of the global clock, a sub-program executes an action based on the information it receives. Since a sub-program can only react to the information it receives, it must behave uniformly at any two states of the system at which the sub-system is identically 'informed'. We call the information known to a sub-system, at a point in its execution history, as the *view* of the sub-system.

- Due to the system being reactive, the behaviour of a distributed program is given by a *computation tree*, which is an edge labelled directed tree with all its edges pointing away from the root. It can be described informally as follows:
  - Each node of the computation tree corresponds to a possible state of the system after a partial run of the distributed program, and the root of the tree corresponds to the start state of the system. Also, the state of any sub-system can be identified easily from the state of the system.
  - Each edge is labelled with a collection of actions, one corresponding to an action of each sub-system. An edge from a node to another denotes a transition of the system between the states represented by the nodes. The branching at a node represents the possible transitions due to the joint execution of a collection of actions, and each branch corresponds to the transition due to a particular input from the environment at that state.
  - The computation tree of a distributed program also satisfies some additional constraints, due to the earlier mentioned requirement that a sub-program must behave uniformly when it has the same view. We call these the 'uniformity constraints' of the computation tree.

The specification of the behaviour of distributed programs consists of two parts:

- *Information Specification*: The access that a sub-system has to the information held by a distributed system, is given by means of an information specification. This is typically given by means of an *architecture*, which is an edge labelled directed graph, with vertices of the graph representing the various sub-systems and labels on the edges representing *variables* shared by the sub-systems. A labelled directed edge from a vertex v to another vertex v', with the edge label containing a variable z, denotes that the sub-system represented by v has access to write on the variable z, and that the sub-system represented by v' has access to read the variable z.

  We note here that there could be other modes of communication,that could be studied in the context of synthesis, but here we consider only communication by shared variables.

- *Execution Specification*: The term execution specification refers to the properties of the computation tree generated by a distributed program. These can be broadly classified into local and global specifications. A global specification refers to assertions on the computation tree of a distributed program, whereas a local specification refers to assertions on the 'behaviour' of the sub-programs. The notion of behaviour for sub-program is also given by a computation tree, in which the nodes correspond to the states of the sub-system and the edges are labelled by the actions performed by the sub-program.

Distributed Synthesis Problem and Winning Strategy Problem The distributed synthesis problem admits a natural correspondence with the *winning strategy problem* for *multi-agent imperfect information games*. This can be explained informally as follows: The input specification for the distributed synthesis problem is transformed to an adversarial game between a *coalition* of agents and an environment-agent. The information specification of the distributed program now becomes a 'action-labelled game graph', where the vertices of the game graph represent the states of the system, and the edges represent the state change due to the execution of actions that label the edge. The execution specification becomes the winning condition for the game. The game tree for this game is given by an 'unravelling' of this game graph, where the nodes of the tree represent execution histories and the 'indistinguishability relation' for an agent relates execution histories that are informationally identical to that agent. A strategy for this game now corresponds to a distributed program, the desired distributed program to be synthesized now corresponds to a winning strategy of such a game.

Much like above, the synthesis problem for reactive centralized systems, called the *classical synthesis problem*, admits a natural correspondence with the *winning strategy problem* for two agent *perfect information games*. A perfect information game is a game in which a single agent plays against the environment-agent, and additionally, throughout the game, both agents receive exact information about the state. In this case the game tree has trivial indistinguishability relations: the indistinguishability relation for each agent only relates the node of the game tree to itself.

We note here that distributed systems specified by architectures only allow modelling of situations where the communication structure between the various sub-systems is fixed beforehand. On the other hand imperfect information games are capable of modelling distributed systems where the communication structure between sub-systems is dynamic.

A Summary of the Distributed Synthesis Problem

Before we begin this section, we remark that the summary presented here is in no way complete, and is intended only to point out the milestones, that in the author's opinion, are the most essential to the study of the distributed synthesis problem. We organize the results in increasing order of generality, with a side-by-side comparison of the synthesis question with the corresponding winning strategy problem.

The synthesis question was first raised in [10] by Alonzo Church, for switching circuits with specification given in restricted recursive arithmetic. In our terminology, this is the classical synthesis problem. In [11] Büchi and Landweber show that the problem for such systems is decidable, against a more general class of specifications called the monadic second-order logic of order. Later in [12] Rabin gave a tree-automata-theoretic solution to this problem.

The solution of Büchi and Landweber involves the reduction of the classical synthesis problem, to a *winning strategy problem* for a perfect information game, whereas the solution given by Rabin, reduces the classical synthesis problem to obtaining a regular tree from a tree-automaton. The ideas from this automata-theoretic solution form the basis of many of the solutions of synthesis problems studied in this tradition.

The next milestone for the winning strategy problem is in [13],[14], in which Reif shows how to solve the winning strategy problem for an imperfect information game with a one-agent coalition and with 'reachability winning condition'. The solution in this case proceeds by a reduction of the imperfect information game to a perfect information game, using ideas similar to that of the subset construction used for determinizing finite-state word automata.

In the synthesis setting, the above game corresponds to reactive distributed systems with a single sub-system, in which the environment has inputs to the system that are unavailable to the sub-system. In [15], Kupferman and Vardi show that the synthesis problem for such systems is solvable, for the specification language of CTL^*. The crucial tool here is the automata-theoretic operation called *narrowing*, that interprets the set of computation trees of distributed programs that satisfy the specifications, as a regular set of trees accepted by a tree-automaton. The desired distributed program is obtained by finding a regular tree accepted by this tree-automaton, and then inverting the interpretation into a distributed program.

The earliest study of multi-agent imperfect information games in its full generality, is in [1], where Peterson and Reif show that the winning strategy problem for multi-agent imperfect information games is unsolvable, even for very simple games and winning conditions. They go on to show that for a class of games called 'hierarchical games', the winning strategy problem is solvable for reachability winning condition. This result was extended to the case of winning conditions that are 'observable parity conditions' by Berwanger et.al. in [16], by means of a generalization of the subset-like construction. Their solution reduces an imperfect information game to a perfect information game. The proof here gives a general criterion, for when this reduction results in a perfect information game that is described by a ‘finite game graph’. They show the solvability of a hierarchical game, by showing that the hierarchical game satisfies this criterion.

The distributed synthesis problem in its full generality was first studied by Pnueli and Rosner in [2]. They show the unsolvability of the synthesis problem for arbitrary architectures with ‘LTL specifications’, and that the synthesis problem is solvable for ‘hierarchical architectures’ with such specifications. Later in [3], Kupferman and Vardi extend this to a larger class of architectures and for global μ-calculus specifications, using the narrowing construction that they had used earlier. The state-of-the-art result in the case of global specifications was given by Finkbeiner and Schewe in [4], in which they give a criterion for architectures called the ‘information forks’ criteria, the absence of which is a necessary and sufficient condition for solving the distributed synthesis problem for μ-calculus specifications. Here again, the essential tool is the automata-theoretic operation of narrowing.

The study of local specifications for the synthesis problem, was introduced by Madhusudhan and Thiagarajan in [5], where they show that for ‘acyclic architectures’, an architecture being a sub-architecture of a ‘flanked-pipeline’ is a necessary and sufficient condition for solving the distributed synthesis problem with local specifications given in linear temporal logics. Later Fridman and Puchala show in [6] that this solution can be extended to the case of ‘flanked-pipelines with feedback’.

Thesis Outline

The main goal of the thesis is to study the winning strategy problem for multi-agent imperfect information games with global or local μ-calculus winning conditions, with an eye towards the distributed synthesis problem. We remark here that the specification language of μ-calculus is at least as expressive as all the specification languages mentioned above.

The focus in this thesis is to obtain purely 'game-based' solutions to the winning strategy problem, by means other than the automata-theoretic solutions. The motivation for this is the following: The problem depends crucially on the view that a sub-system has in relation to the exact state of the system. While the automata-theoretic approach implicitly considers this aspect, the game-based solution explicates the relationship between views, by means of the indistinguishability relation on the game trees. This allows us to build a framework that gives fundamental reasons for the solvability of the corresponding winning strategy problems, based on the structure of indistinguishability relations. This allows for a finer classification of the solvability landscape, as opposed to considering structure of the architecture. Another advantage of the game approach is the robustness it offers to slight changes in the problem statement; as an example of this, we will show that the synthesis of non-determinsitic and determinstic distributed programs can be argued simultaneously in many cases.

Next we summarize the contents the thesis:

- In Chapter 2, we introduce the winning strategy problems studied in this thesis. We begin by introducing the necessary preliminaries to state the winning strategy problems, and then show the translation of the distributed synthesis problem to a winning strategy problem on imperfect information games. We conclude the chapter with a summary of known results for the distributed synthesis problem.
- In Chapter 3 we study a class of imperfect information games called *games with recurring common knowledge of state*, and show that such games are solvable for winning conditions given by 'priority labelling' of the game graph. This is intended as a departure from the class of hierarchical games, which are the only previously known case of multi-agent imperfect information games that admit a solution to the winning strategy problem.

- The remaining chapters study a generic approach to solving imperfect information games, called the *retraction approach*. Chapter 4 introduces the idea behind this approach, and equips us with some concepts and tools to solve the winning strategy problem. The key idea here is the design of a class of structures, called 'witnesses', that witness the existence of a winning strategy in the game. Then we discuss a generic operation called 'retraction', that transforms witnesses to possibly smaller witnesses. We show that if a class of witnesses of a game, called *canonical witnesses*, admit retraction to small witnesses, then it is a sufficient criteria for solving the winning strategy problem. We with a description of how the distributed synthesis problem can be solved by solving the corresponding winning strategy problem.

- In Chapter 5 we consider the case of global specifications on architectures that admit *weak-informedness ordering* and *informedness ordering* on its agents. These classes are known to have a decidable distributed synthesis problem, and we give new proofs for these via the retraction approach. We show that these architecture classes correspond to games that contain no *uniform-determinsitic fork-triples* and *fork-triples* respectively, and that canonical witnesses obtained from such games belong to a class of witnesses called *modular witnesses*. To solve the corresponding winning strategy problem, we show that modular witnesses admit retractions to small witnesses.

- In Chapter 6 we introduce the class of *weak-broadcast architectures* and *broadcast architectures*, and show that for the case of local specifications, these architecture have a decidable distributed synthesis problem. We note here that these architecture classes subsume the case of flanked pipelines and flanked pipelines with feedback. The solution for this proceeds in two steps: first we show that canonical witnesses for games corresponding to these classes can be *factorized* into simpler witnesses, and second, we show that such witnesses admit retractions to small witnesses by means of the retractions of their 'factors'.

Chapter 2

The Winning Strategy Problem

In this chapter we formally define a multi-agent imperfect information game, and the winning strategy problem. Specifically, we describe how imperfect information games and winning conditions of games are specified via game-graphs and μ-automata. Later, we describe in detail the relationship between the the distributed synthesis problem and the winning strategy problem, and conclude the chapter by presenting some known results from the literature that are relevant to the study here.

2.1 Preliminaries

Standard Notation : Any totally-ordered set I may be referred to as an *index set* and its elements called *indices*. $\{X_i\}_{i\in I}$ denotes a collection of sets X_i indexed by some index set I. $(x_i)_{i\in I}$ denotes a tuple $(x_1,x_2,..)$ indexed by I and $\prod_{i\in I} X_i$ denotes the product $X_1 \times X_2 \times ..$ indexed by I. For any tuple x indexed by a set I, we use $x^{\downarrow i}$ to denote its i^{th} component. $\mathscr{P}(X)$ and 2^X denote the power-set of the set X. Given sets X,Y we denote a function f from X to Y by $f : X \to Y$ and a partial function f' from X to Y by $f' : X \hookrightarrow Y$. Let $\mathbb{N}$ denote the set of natural numbers and for a natural number n, let $[n]$ denote the set $\{1,2,..,n\}$. For a binary relation $Y \subseteq X \times X$, we use xY to denote the set $\{x' \mid (x,x') \in Y\}$.

For a function $f : X \to Z$ and a subset $X' \subseteq X$, $f(X')$ denotes the set $\{f(x) \mid x \in X'\}$. For functions f, g, we denote their composition by $f \circ g$ and define it as $f \circ g(x) = g(f(x))$ for all x. Given an equivalence relation $Y \subseteq X \times X$, we denote the equivalence classes generated by Y as X/Y and the equivalence class containing an element $x \in X$ by $[x]_Y$.

2.1.1 ω-Words

Let us fix a finite set of *symbols* that we call an *alphabet*, and denote it by Σ. A *word* over Σ is a finite sequence of symbols in Σ. The set of all words over Σ is denoted by Σ^* and a *word language* over Σ is a subset of Σ^*. Similarly, an ω-*word* (read as omega-word) over Σ is an infinite sequence of symbols in Σ. The set of all ω-words over Σ is denoted by Σ^ω and an ω-*word language* over Σ is subset of Σ^ω.

ω-Word Automata An ω-*word automaton* is given by a tuple $(Q, \Sigma, \Delta, \mathsf{Acc}, q_\varepsilon)$, where Q is a finite set of states with q_ε as the start state, Σ is the alphabet, $\Delta \subseteq Q \times \Sigma \times Q$ is the *transition relation* and Acc is the *acceptance condition*. We introduce only the minimum ideas associated with ω-automaton, and refer the reader to [17] for a comprehensive treatment of these.

The ω-word automaton is qualified as *deterministic* if the relation $\Delta \subseteq Q \times \Sigma \times Q$ is a partial function[1] $\Delta : Q \times \Sigma \hookrightarrow Q$, otherwise it is qualified as *non-deterministic*.

A *run* of an ω-word (or a word) $w_1 w_2..$ on the ω-word automaton $\mathscr{A}$, is a sequence $q_0 q_1..$ that satisfies $q_0 = q_\varepsilon$, and $(q_i, w_{i+1}, q_{i+1}) \in \Delta$ for all index $i, i+1$ of the ω-word.

An ω-word automaton is parametrized by various kinds of acceptance conditions, and we list some of them below. First we introduce some notation. For an ω-word $\widetilde{x} = x_1 x_2..$ over an alphabet X, let $\mathsf{Oc}(\widetilde{x})$ denotes the set of elements in X that appear in the sequence and let $\mathsf{Inf}(\widetilde{x})$ denote the set of elements in X that appear infinitely often in the sequence.

[1]Note that this is different from the usual; usually this is called an *incomplete* deterministic automata

- *Büchi/Co-Büchi Automaton*: Both *Büchi automaton* and *Co-Büchi automato*n are given by an ω-word automaton $\mathscr{A} = (Q, \Sigma, \Delta, F, q_\varepsilon)$ where $F \subseteq Q$.
  A run $q_0 q_1 ..$ of the ω-word $w_1 w_2 ..$ on the automaton $\mathscr{A}$ is said to be an *acceptance run* of

  - a Büchi automaton $\mathscr{A}$, if $\mathsf{Inf}(q_0, q_1, ..) \cap F \neq \emptyset$.
  - a Co-Büchi automaton $\mathscr{A}$, if $\mathsf{Inf}(q_0, q_1, ..) \cap F = \emptyset$.

  Note that a run is an accepting run for a Büchi automaton $\mathscr{A}$, if and only if, the run is not an accepting run for a Co-Büchi automaton $\mathscr{A}$.

- *Parity Condition*: A *parity automaton* is given by an ω-word automaton $\mathscr{A} = (Q, \Sigma, \Delta, \Omega, q_\varepsilon)$ where $\Omega : Q \rightarrow P$ is a function that maps states of the automaton to the set of *priorities* P such that $P = \{1, 2, .., |P|\}$.
  A run $q_0 q_1 ..$ of the ω-word $w_1 w_2 ..$ on the automaton $\mathscr{A}$ is said to be an *acceptance run* of the Parity automaton $\mathscr{A}$, if the priority sequence $\Omega(q_0)\Omega(q_1) ..$ satisfies the *parity condition*, that is, the minimum priority in $\mathsf{Inf}(\Omega(q_0)\Omega(q_1) ..)$ is even.

In all the above cases, a word is said to be *accepted* by an ω-word automaton with a particular acceptance condition, if there exists an acceptance run of the word on the respective ω-word automaton. We call the set of words accepted by an ω-word automaton as the ω-word language *recognized* by the ω-word automaton.

Theorem 2.1.1 (Chapter 1 in [17])**.**

1. *Büchi automata and parity automata are equivalent in expressive power, that is, they recognize the same ω-word languages.*
2. *Deterministic parity automata recognize the same ω-word languages as Büchi automata.*

Theorem 2.1.2 ([18]). *For every Büchi automaton $\mathscr{A}$ with n states, there exists a deterministic parity automaton with n^{2n+2} states and $2n$ priorities, that recognizes the same ω-language as $\mathscr{A}$.*

The next theorem is a folklore result in automata theory, but for the sake of completeness we prove it here.

Theorem 2.1.3.

1. *For every parity automaton with n states and k priorities, there exists a Büchi automaton with at most $n(k+1)$ states.*
2. *For every deterministic parity automaton $\mathscr{A}$ with n states, there exists a deterministic parity automaton with n states that recognizes the complement language recognized by $\mathscr{A}$.*

Proof. Towards the first part of the theorem, consider a parity ω-word automaton $\mathscr{A}$. The desired Büchi automaton is constructed as follows: Consider k copies of the automaton $\mathscr{A}$, one corresponding to each priority, and an additional copy that we call the initial copy. We denote the copy corresponding to a priority j by $\mathscr{A}_j$ and the initial copy by $\mathscr{A}_{in}$. From each copy $\mathscr{A}_j$, we remove all the states with odd priorities that are less than or equal to j, and include the states with priority j as the final states of the Büchi automaton. We leave intact all the transitions in each copy $\mathscr{A}_j$. Additionally, we add transitions so that at any state q in the initial copy, if there exists an outgoing transition (q,a,q'), then for every copy $\mathscr{A}_j$, we add an outgoing transition (q,a,q'_j), where q'_j denotes the corresponding copy of the state q' in the copy $\mathscr{A}_j$. This completes the construction of the desired Büchi automaton.

It remains to show that the language recognized by both these automata are identical. Firstly, note that if there exists an acceptance run of an ω-word on a parity automaton, then there exists an acceptance run of this ω-word on the constructed Büchi automaton. Towards this, consider an accepting run of the parity automaton $\mathscr{A}$. Note that by definition of an acceptance run of a parity automaton, it must be the case that there exists a suffix of this run and an even priority j that satisfy the following: the priorities appearing in this suffix are greater than or equal to j and the priority j appears infinitely often in this suffix. We call this the *good* suffix of an accepting run. Now consider a run of the Büchi automaton that mimics the run of parity automaton in the initial copy until the beginning of the good suffix, at which point the Büchi automaton jumps to the copy of the current state of the automaton in the copy $\mathscr{A}_j$ and mimics the remaining run of the parity automaton in this copy. By construction of the Büchi automaton, it follows that this is an accepting run.

For the other direction, consider an acceptance run of the Büchi automaton. By construction of the automaton, it must be the case that every run eventually settles in some copy $\mathscr{A}_j$ or the initial copy $\mathscr{A}_{in}$. Moreover, due to the way in which final states of the Büchi automaton are constructed, it must be the case that an accepting run settles in some copy $\mathscr{A}_j$ with an even j, and the run is accepting because it visits infinitely many states of priority j in the copy $\mathscr{A}_j$. By mimicking this run in the parity automaton, we obtain an accepting run on the parity automaton, which has a suffix such that the minimum priority seen infinitely often along this suffix is j.

Towards the second part of the theorem, consider a deterministic parity automaton that has priority labelling Ω, and construct a parity automaton that is identical to the original parity automaton, but with a priority labelling Ω', which is obtained by incrementing the priority assignment on the states by one. Observe that for any run $q_0, q_1, ..$ in the original parity automaton, the priority sequence $\Omega(q_0)\Omega(q_1)..$ satisfies the parity condition, if and only if, the priority sequence $\Omega'(q_0)\Omega'(q_1)..$ does not satisfy the parity condition. Since both parity automata have identical transition relations, and are deterministic, every ω-word generates a unique run in both the parity automaton, if it exists. Therefore an ω-word is accepted by the original automaton, if and only if it is not accepted by the above constructed automaton. This shows that the language recognized by the automaton constructed here is the complement of the language of the original automaton. □

2.1.2 Multi-Agent Transition Systems

The objects that are central to our analyses of the winning strategy problem for imperfect information games are structures called 'multi-agent transition systems', and these are defined next. These are familiar objects in many studies that involve multi-agent systems [19],[20], but their use is rare for the question of distributed synthesis problem.

Labelled Transition System An *A-transition system*, abbreviated as A-TS, is a structure (S,R,s_ε) where A denotes the set of *actions* of the system, the set S denotes the set of *states* of the system with s_ε as the *start state*, the relation $R \subseteq S \times A \times S$ denotes the *transition relation* and a tuple $(s,a,s') \in R$ denotes an *a-transition* from state s to s' on an *action* $a \in A$.

A *$\{B_j\}_{j\in J}$-labelled A-transition system*, abbreviated as $(\{B_j\}_{j\in J},A)$-LTS, is a structure $(S,R,\{\nu\}_{j\in J},s_\varepsilon)$ that is an expansion of the A-TS (S,R,s_ε) with *labelling functions* $\nu_j : S \to B_j$, one for each $j \in J$. We refer to elements in each B_j as *labels*.

Next we define some notation and terminology associated with a A-TS $\mathscr{S} = (S,R,s_\varepsilon)$.

- We say that a transition $(s_1,a,s_2) \in R$ is an *incoming* transition at a state s, if $s = s_2$ holds and an *outgoing* transition at a state s, if $s = s_1$.

  By an *a-transition*, we mean a transition labelled with an action a.

- A *path* in the TS $\mathscr{S}$ is an infinite alternating sequence $s_0a_1s_1a_2...$ of states and joint actions such that $(s_k,a_{k+1},s_{k+1}) \in R$ for all indices $k \geq 0$. By a *prefix* of a path, we mean a prefix of the path that ends at a state, as opposed to ending at an action. By the *length* of a path $\tau = s_0a_1s_1\ldots a_ks_k$, we mean the index k. We also call a path $\widetilde{s}$ to be a *prolongation* of a path $\widetilde{s}'$, if the path $\widetilde{s}'$ is a strict prefix of $\widetilde{s}$.

  For a path $\widetilde{s}$, we use $\mathsf{start}(\widetilde{s})$ to denote its start state and use $\mathsf{end}(\widetilde{s})$ to denote its end state, if it exists. A state s_2 is said to be *reachable* from s_1, if there exists a path $\widetilde{s}$

such that $\mathsf{start}(\widetilde{s}) = s_1$ and $\mathsf{end}(\widetilde{s}) = s_2$.

- A TS is qualified as a *acyclic* if there exist no 'simple cycles' in the TS, and every state in the TS is reachable from the start state. A *simple cycle* is a path $s_0a_1...a_ms_m$ such that $s_0 = s_m$ and no other state except s_0 repeats. A TS is qualified as a *tree*, if it is acyclic, and additionally, if every state in the TS, except the start state, has exactly one incoming transition. We call the start state of an acyclic TS, a *root*, and a state with no outgoing transitions as a *leaf*.

- We qualify $\mathscr{S}$ as *deterministic*, if for any state $s \in S$, any pair of distinct outgoing transitions $(s,a_1,s_1),(s,a_2,s_2) \in R$ at s, satisfy $a_1 \neq a_2$. We qualify $\mathscr{S}$ as *non-terminal*, if every state in S has some outgoing transition.

- A substructure of $\mathscr{S}$ induced by a set of states $S' \subseteq S$, is called *strategic*, if $s_\varepsilon \in S'$, and additionally, for any state $s \in S'$ and joint action $a \in A$, $sR_a \cap S' \neq \emptyset$ implies $sR_a \subseteq S'$.

- For any joint action $a \in A$, we use R_a to denote the relation $\{(s,s') \in S \times S \mid (s,a,s') \in R\}$. For a state $s \in S$, we call the set of states sR_a as the *a-neighbourhood* of s. For any state $s \in S$, the *neighbourhood* of s is the partial function $R(s) : A \rightharpoonup \mathscr{P}(S)$ defined as $R(s)(a) = sR_a$ for all $s \in S$.

Multi-Agent Transition System An $\{A_i\}_{i\in[n]}$*-multi-agent transition system*, abbreviated as $\{A_i\}_{i\in[n]}$-MaTS, is a structure $(S,R,\{\sim_i\}_{i\in[n]},s_\varepsilon)$ where each set A_i denotes the *action set* of agent i, each equivalence relation $\sim_i \subseteq S \times S$ denotes the *indistinguishable relation* for agent i with a tuple $(s,s') \in \sim_i$ denoting that s,s' are *indistinguishable* for i, and the structure (S,R,s_ε) is a A-TS, where $A = \prod_{i\in[n]} A_i$ is called the set of *joint actions*.

A $\{B_j\}_{j\in J}$*-labelled* $\{A_i\}_{i\in[n]}$*-transition system with indinstinguishability relations*, abbreviated as $(\{B_j\}_{j\in J},\{A_i\}_{i\in[n]})$-$\mathsf{MaLTS}$, is a structure $(S,R,\{\sim_i\}_{i\in[n]},\{\nu_j\}_{j\in J},s_\varepsilon)$ that is an expansion of the $\{A_i\}_{i\in[n]}$-MaTS $(S,R,\{\sim_i\}_{i\in[n]},s_\varepsilon)$ with *labelling functions* $\nu_j : S \to B_j$, one for each $j \in J$.

We say that an $\{A_i\}_{i\in[n]}$-MaTS $\mathscr{S} = (S, R, \{\sim_i\}_{i\in[n]}, s_\varepsilon)$

- is *uniform*, if
  - for any states $s_1, s_2 \in S$ and agent $i \in [n]$, if $s_1 \sim_i s_2$ holds, then $\{a^{\downarrow i} \in A_i \mid s_1 M_a \neq \emptyset\} = \{a^{\downarrow i} \in A_i \mid s_2 M_a \neq \emptyset\}$, and
  - for any state $s \in S$, $\{a \in A \mid sM_a \neq \emptyset\} = \prod\limits_{i\in[n]} \{a^{\downarrow i} \mid sM_a \neq \emptyset\}$.
- has *perfect-recall*, if for any transitions $(s_1, a, s_2), (s_1', a', s_2') \in R$ and agent $i \in [n]$, $s_2 \sim_i s_2'$ implies that $s_1 \sim_i s_2$.

Intuitively, the first condition asserts that for each agent in the coalition, the set of actions prescribed by an agent at indistinguishable histories is identical. The second condition above is a sanity check for whether the joint actions at a state allow all possible combinations of actions prescribed by the agents.

We note that a definition associated with a structure may also be invoked for expansions of the structures, for example a path of LTS refers to the path of the TS ‘underlying’ the LTS. We use the term ‘underlying’ to refer to the various reducts of structures; for example given a LTS $(S, R, \{\sim_i\}_{i\in[n]}, \{\nu_j\}_{j\in J}, s_\varepsilon)$, we call the LTS $(S, R, \{\nu_j\}_{j\in J}, s_\varepsilon)$ as the LTS *underlying* the MaLTS. Another convention we follow is that whenever the parameters being referred to in a definition are clear from the context, then we discard them, for example a $(\{B\}, \{A_i\}_{i\in[n]})$-$\mathsf{MaLTS}$ will be referred to as a MaLTS.

μ-Calculus and μ-Automata The μ-calculus is a modal logic with fixed-point operators, that are interpreted on Kripke models. A LTS $(S, R, \nu, s_\varepsilon)$ may be seen as a Kripke model with the state set S denoting the set of possible worlds, the state s_ε as the current world and each of the relations in $\{R_a \mid a \in A\}$ denoting the accessibility relations for different modalities. For the purposes here, instead of the logic of μ-calculus, we use μ-automata to specify properties of LTS's, since μ-calculus and μ-automata have the same expressive power on the class of LTS's [21].

A *μ-automaton* on (B,A)-LTS's is given by a tuple $(Q,A,B,\delta,\Omega,q_\varepsilon)$, where Q is a finite set of states with q_ε as the start state, A is the set of joint actions, $\delta : Q\times B \to \mathscr{P}(A\to\mathscr{P}(Q))$ is the *transition function* and $\Omega : Q \mapsto P$ is a function that assigns states to *priorities* in $P=\{1,2,..,|P|\}$. We say that a state $q\in Q$ has priority p, if $\Omega(q)=p$.

We say that a (B,A)-LTS $\mathscr{S}=(S,R,\nu,s_\varepsilon)$ is *accepted* by the μ-automaton $\mathscr{Q}=(Q,A,B,\delta,\Omega,\nu_\varepsilon)$ via the *acceptance run $\mathscr{S}'$*, if $\mathscr{S}'$ is a $(\{S,Q\},A)$-LTS $(S',R',\{f_S,f_Q\},s'_\varepsilon)$ that satisfies the following:

- $f_S(s'_\varepsilon)=s_\varepsilon$ and $\forall s'\in S',\forall a\in A : f_S(s'R'_a)=f_S(s')R_a$,
- $f_Q(s'_\varepsilon)=q_\varepsilon$ and $\forall s'\in S' : \{(a,f_Q(s'R'_a)) \mid a\in A\}\in\delta(f_Q(s'),\nu(f_S(s')))$, and
- for every path $s'_0a_1s'_1a_2..$ in $\mathscr{S}'$, the priority sequence $\Omega(f_Q(s'_0)),\Omega(f_Q(s'_1)),..$ satisfies the parity condition.

Intuitively, the first condition for the acceptance run asserts that the function f_S is a 'bisimulation' from the TS (S',R',s'_ε) onto the TS (S,R,s_ε). The second condition asserts that for every state $s\in S'$, the labelling ℓ_Q of the states in the neighbourhood of s are 'compatible' with the transition function δ in the way described above.

We note that in the original presentation of μ-automaton in [21], the acceptance run is also required to be a tree. It is not necessary to add this condition, since the 'tree unravelling' of an acceptance run as given here, continues to be an acceptance run.

Next we define the notion of 'strong acceptance' by a μ-automaton. We say that $\mathscr{S}$ is a *strongly accepted* by $\mathscr{Q}$ via a *strong acceptance run f_Q*, if f_Q satisfies the following:

- $f_Q(s_\varepsilon)=q_\varepsilon$ and $\forall s\in S : \{(a,f_Q(sR_a)) \mid a\in A\}\in\delta(f_Q(s),\nu(s))$, and

- for every path $s_0a_1s_1a_2..$ in $\mathscr{S}$, the priority sequence $\Omega(f_Q(s_0)), \Omega(f_Q(s_1)), ..$ satisfies the parity condition.

Note that a strong acceptance run is special case of an acceptance run $(S', R', \{f_S, f_Q\}, s'_\varepsilon)$, in which the labelling f_S is the identity map on states of S.

Tree Automaton A (B,A)-LTS is called a (B,A)-*ranked tree*, if it is a tree, and if for every joint action $a \in A$, the existence of an outgoing a-transition at a state in the LTS, implies the existence of a unique outgoing a-transition at the state.

An *alternating tree automaton* [22],[17] is a μ-automaton, with its acceptance condition additionally requiring that the LTS input to the automaton be a ranked tree.

A *tree automaton* is a special case of an alternating tree automaton where the acceptance criterion is that of strong acceptance, that is, a ranked tree is said to be *accepted* by a tree automaton $\mathscr{Q}$, if it is strongly accepted by the alternating tree automaton $\mathscr{Q}$. Since the notion of acceptance here is that of strong acceptance, and since the LTS's being accepted are ranked trees, we may assume without loss of generality that any tree automaton $(Q, A, B, \delta, \Omega, q_\varepsilon)$ satisfies the property that for every $q \in Q$, $b \in B$, $f \in \delta(q,b)$ and $a \in A$, we have $|f(a)| \leq 1$.

Theorem 2.1.4 ([23])**.** *For every alternating tree-automaton $\mathscr{A}$ with n states and k priorities, there exists a tree automaton $\mathscr{A}'$ with $2^{\mathscr{O}(nd\ log(nd))}$ states and $\mathscr{O}(nd)$ priorities such that, a ranked tree is accepted by $\mathscr{A}$, if and only if, it is accepted by $\mathscr{A}'$.*

2.2 Imperfect-Information Games

An *imperfect information game* between a *coalition* of agents $[n]$ and the environment, is given by a tuple $(\mathscr{T}, W)$ consisting of two parts, a 'game tree' $\mathscr{T}$ and a 'winning condition' W for 'strategies' on the game tree.

Game Trees A *game tree* for the coalition $[n]$ (against the environment) is an $\{A_i\}_{i\in[n]}$-MaTS $\mathscr{T} = (S, M, \{\sim_i\}_{i\in[n]}, s_\varepsilon)$ that is a tree, where S denotes the *states* of the game with s_ε denoting the *start state*, each $\sim_i \subseteq S \times S$ is an equivalence relation that denotes the *indistinguishability relation* of agent i, the transition relation $M \subseteq S \times A \times S$ denotes the *move* relation on the game tree on *joint actions* in A. Moreover, for every state $s \in S$ and joint action $a \in A$, there exists some outgoing a-transition at s.

Next, we explain how a game proceeds. A *play* in a game begins at the start state s_ε and proceeds as follows. At a state s during the play, each agent in the coalition independently chooses a subset A'_i from its set of actions A_i. The environment responds by non-deterministically choosing an action a_i from this set A'_i, resulting in a joint action $(a_i)_{i\in[n]}$, and chooses the next state s' that satisfies $(s, a, s') \in M$. The current state of play is now updated to s' and this process repeats. Formally, a *play* is defined as a path in the game tree $\mathscr{T}$ that begins at the start state. A finite prefix of a play is called a *history*.

Along a play, agents may have different information about the play. The role of indistinguishability relations is to denote the information available to an agent at a history, by means of an equivalence relation. An equivalence $s \sim_i s'$ in the game tree denotes that the agent i has identical information at the states s and s', and they are therefore *indistinguishable* for agent i. The equivalence classes induced by the indistinguishability relation $\sim_i$ of agent i are called the *information sets* of agent i. If all the information sets of an agent are singletons, then we say that the agent i has *perfect information*, otherwise we say that the agent i has *imperfect information*. A *perfect information game* is one where all agents have perfect information.

There are two equivalent formalisms for defining the strategy of an imperfect information game, namely, 'coalition strategy' and 'joint strategy'.

- A *coalition strategy* on the game tree $\mathscr{T}$, is a tuple of functions $(\varsigma_i)_{i\in[n]}$, one for each agent in $[n]$, where each $\varsigma_i : S/\sim_i \to (2^{A_i} \setminus \emptyset)$ gives the *strategy of agent i*, that maps information sets of agent i to a non-empty set of actions in 2^{A_i}.

- A *joint strategy* on the game tree $\mathscr{T}$, is a function $\sigma : S \to (2^A \setminus \emptyset)$, such that
  - for any states $s_1, s_2 \in S$ and agent $i \in [n]$, if $s_1 \sim_i s_2$ holds, then $\{a^{\downarrow i} \in A_i \mid a \in \sigma(s_1)\} = \{a^{\downarrow i} \in A_i \mid a \in \sigma(s_2)\}$, and
  - for any state $s \in S$, $\{a \in A \mid a \in \sigma(s)\} = \prod\limits_{i \in [n]} \{a^{\downarrow i} \in A_i \mid a \in \sigma(s)\}$.

It is not difficult to see that these two notion of strategy are equivalent. Towards arguing this, consider the following translations.

- Given a coalition strategy $(\varsigma_i)_{i \in [n]}$, the corresponding joint strategy σ is defined for any state $s \in S$ as follows: $\sigma(s) = \{(a_i)_{i \in [n]} \in A \mid \forall i \in [n] : a_i \in \varsigma_i([s]_{\sim_i})\}$. It is easy to see that σ satisfies both the conditions necessary for a joint strategy.
- Given a joint strategy σ, the corresponding coalition strategy $(\varsigma_i)_{i \in [n]}$ is defined for any agent i and information set $S' \in S/{\sim_i}$ as follows: $\varsigma_i(S') = \{a^{\downarrow i} \in A_i \mid \exists s \in S' : a \in \sigma(s)\}$.

We consider the notions of coalition strategy and joint strategy to be equivalent for the following reason: the translation of a coalition strategy to a joint strategy and a further translation of this joint strategy back to a coalition strategy, results in isomorphisms between the strategies of agents. Symmetrically, the translation from a joint strategy to a coalition strategy back to a joint strategy, also results in an isomorphism. This shows that there is no loss of structure when translating one to the other, and therefore, we may use these two notions of strategy inter-changeably.

We qualify a coalition strategy $(\varsigma_i)_{i \in [n]}$ as *deterministic*, if every for every agent $i \in [n]$ and information set $S' \in S/{\sim_i}$, the action-set $\varsigma_i(S')$ is a singleton set. Similarly, we qualify a joint strategy σ as *deterministic*, if every for every agent $i \in [n]$ and state $s \in S$, the action-set $\sigma(s)$ is a singleton set. Note that the translations above translate a deterministic coalition strategy to a deterministic joint strategy, and vice-versa.

Winning Condition A *winning condition* for an imperfect information game is a property of joint strategies on the game tree. We qualify a joint strategy as *winning*, if it satisfies the winning condition.

The *winning strategy problem* for a class $\mathscr{G}$ of imperfect information games is the following: Is there an algorithm that, given an imperfect information game $(\mathscr{T}, W) \in \mathscr{G}$, finds a winning joint strategy for $(\mathscr{T}, W)$ if it exists?

Additionally, if we require that the winning joint strategy be deterministic, then we call it the *deterministic winning strategy problem.*

Towards an algorithmic study of the winning strategy problem, we consider next some finite presentations of game trees, winning conditions and strategies.

Specifying Imperfect Information Games Here we consider game trees given by finite game graphs, which are similar to the 'concurrent game structures' introduced in [24]. In a game graph, each state of the game graph is associated with a tuple of 'local states', one for each agent in the coalition, and on a transition into a state, agents are only informed of the transition into their respective local state. Formally, a *game graph* for a coalition $[n]$ is given by a A-TS $G = (V, E, v_\varepsilon)$, where

- $V \subseteq \prod_{i \in [n]} V_i$ is a finite set of *states* of the game graph, with each V_i denoting the *local state* of the agent i and v_ε denoting the *start state*.
- $E \subseteq V \times A \times V$ denotes the *move* relation of the game graph labelled with the set of joint actions A.

Additionally, for each $v \in V$ and $a \in A$, there is some transition $(v, a, v') \in E$.

A play in a game specified by the game graph G begins at the initial state $v_0 = v_\varepsilon$ and proceeds in rounds. In the first round, each agent i chooses a set of actions $A'_i \subseteq A_i$ at the current state v_0, then environment chooses a joint action $a \in \{(a_i)_{i \in [n]} \mid \forall i \in [n] : a_i \in A'_i\}$

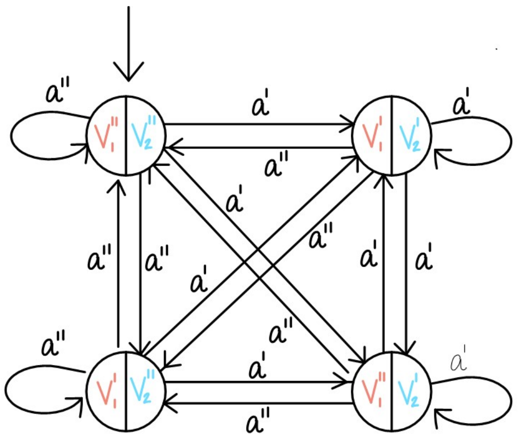


Figure 2.1: A game graph G.

G is a game graph for coalition $\{1,2\}$, with $V_1 = \{v_1', v_1''\}, V_2 = \{v_2', v_2''\}$ giving the local states of $1,2$ respectively, joint action-set $\{a', a''\}$ and with (v_1'', v_2'') as its start state.

and a successor state v_1 reachable via a move $(v_0, a, v_1) \in E$, where $a^{\downarrow i} = a_i$ for every agent i. On moving into the state v_1, each agent i receives information about its local state $v_1^{\downarrow i}$, and the play proceeds to the next round. The next round operates similarly at the next 'current state' v_1, and the play progresses. Formally, a *play* of the game graph G is a maximal path in G that begins at the start state, and a *history* is any finite prefix of a play.

The game tree *specified* by a game graph $G = (V, E, v_\varepsilon)$ is given by the MaTS $\mathscr{T}_G = (H, M, \{\approx_i\}_{i \in [n]}, v_\varepsilon)$ defined as follows:

- H is the set of histories of G and the history v_ε is the start state.
- For any histories $\tau_1, \tau_2 \in H$, joint action a and agent $i \in [n]$, we have
  - $(\tau_1, a, \tau_2) \in M$, if and only if, τ_2 is the prolongation $\tau_1 a v$ of the history τ_1, for some state $v \in \mathsf{end}(\tau_1) E_a$.
  - $\tau_1 \approx_i \tau_2$, if and only if, $\mathsf{view}_i(\tau_1) = \mathsf{view}_i(\tau_2)$,
    where $\mathsf{view}_i(\tau)$ denotes the *information available to agent i at a history τ*, defined inductively as follows:

- ∗ For the trivial history v_ε, we define $\mathsf{view}_i(v_\varepsilon) := v_\varepsilon^{\downarrow i}$, and
- ∗ For a history $\tau a v$, we define $\mathsf{view}_i(\tau a v) := \mathsf{view}_i(\tau) a^{\downarrow i} v^{\downarrow i}$.

Specifying Winning Conditions The term *winning conditions* is used to refer to properties of joint strategies, and the properties studied here are properties of structures called 'strategy trees', that are derived from joint strategies. We define this next.

A *joint strategy* on the game graph $G = (V, E, v_\varepsilon)$ is a joint strategy on the corresponding game tree $\mathscr{T}_G$. We say that a play (or a history) $\pi = v_0 a_1 v_1 a_2 ..$ in the game graph G *follows* a joint strategy σ, if $a_{k+1} \in \sigma(v_0 a_1 .. a_k v_k)$ for any index $k \in \mathbb{N}$ strictly less than the length of π.

An *extended strategy tree* of a joint strategy σ, denoted by $\mathscr{E}\mathscr{T}_G^\sigma$, is given by a $(V, \{A_i\}_{i\in[n]})$-MaLTS $(H^\sigma, M^\sigma, \{\approx_i^\sigma\}_{i\in[n]}, \nu, v_\varepsilon)$, where

- H^σ denotes the set of histories in G that begin at v_ε and follow σ,
- $(H^\sigma, M^\sigma, \{\approx_i^\sigma\}_{i\in[n]}, v_\varepsilon)$ is the substructure of $\mathscr{T}_G$ induced by H^σ, and
- $\nu(\tau) = \mathsf{end}(\tau)$ for every history $\tau \in H^\sigma$.

We call the LTS underlying the extended strategy tree $\mathscr{E}\mathscr{T}_G^\sigma$ as the *strategy tree* of a joint strategy σ, and we denote it by $\mathscr{T}_G^\sigma$.

Winning conditions can be classified into 'global' or 'local' depending on whether they make assertions on the states V of the game graph or on the local states $\{V_i\}_{i\in[n]}$. We state here the winning conditions considered in this thesis, and their corresponding global and local formulations:

- **μ-Automaton Winning Conditions**: For a *global winning condition given by a μ-automaton $\mathscr{Q}$*, we say that a joint strategy σ on the game graph G is *winning*, if the strategy tree of the joint strategy σ, is strongly accepted by $\mathscr{Q}$.

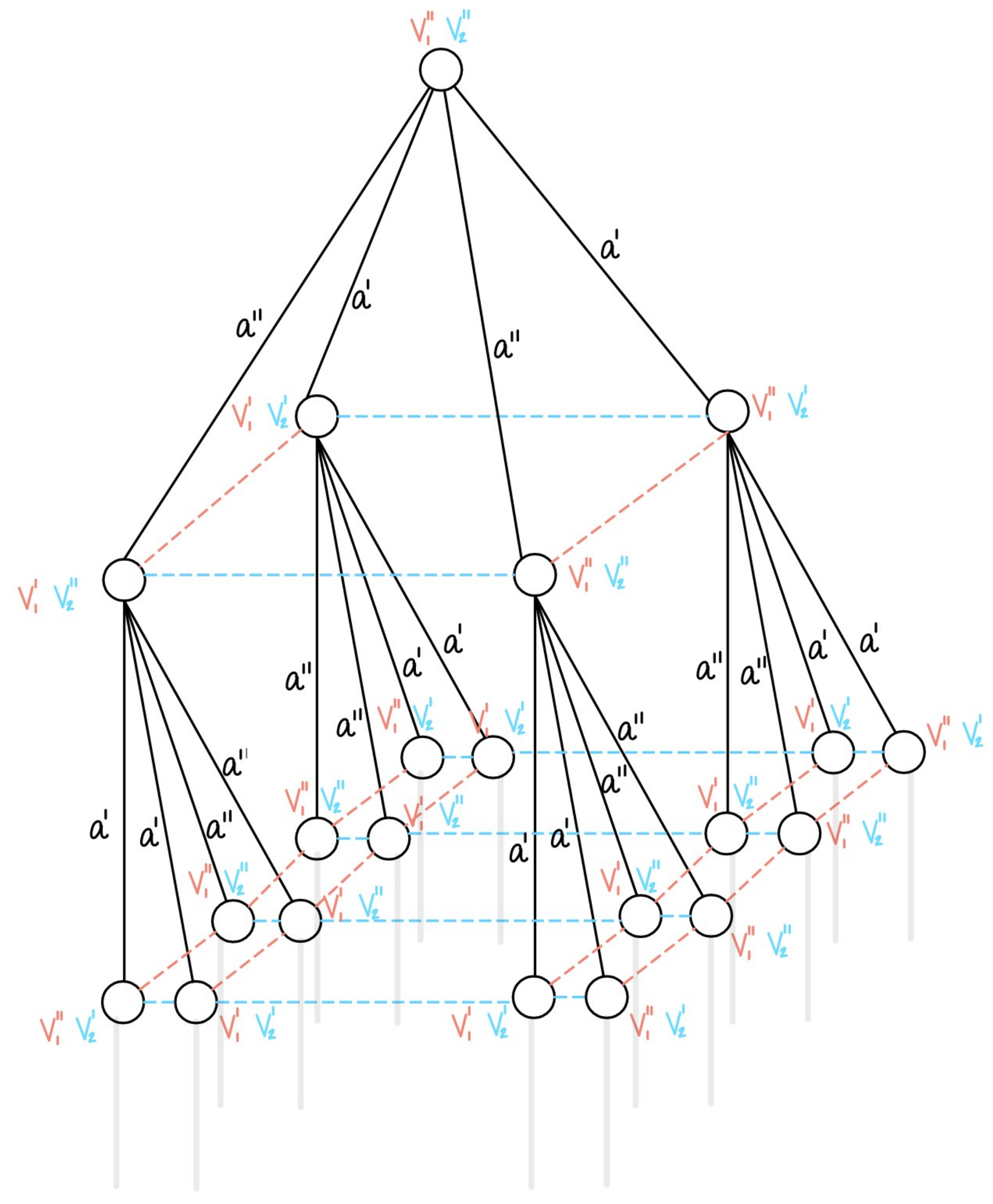


Figure 2.2: The game tree specified by G for histories of length at most 2.

The indistinguishability relation of agents $1,2$ *are respectively denoted by red and blue dashed-edges in the tree. Here for clarity, we draw only some of these dashed-edges, and the indistinguishability relation of an agent* $i \in \{1,2\}$ *is obtained by the reflexive-transitive closure of the edges for i.*

- **ω-Automaton Winning Conditions**:

  - For a *global winning condition given by an ω-automaton $\mathscr{Q}$*, we say that a joint strategy σ on the game graph G is *winning*, if for any play $v_0 a_1 v_1 a_2 v_2 \dots$ that follows σ, the ω-word $v_0 a_1 v_1 a_2 v_2 \dots$ is accepted by the ω-automaton $\mathscr{Q}$.
  - For a *local winning condition given by ω-automata $\{\mathscr{Q}_i\}_{i\in[n]}$*, we say that a joint strategy σ on the game graph G is *winning*, if for any agent $i \in [n]$ and for any play $v_0 a_1 v_1 a_2 v_2 \dots$ that follows σ, the ω-word $v_0^{\downarrow i} a_1^{\downarrow i} v_1^{\downarrow i} a_2^{\downarrow i} v_2^{\downarrow i} \dots$ is accepted by the ω-automaton $\mathscr{Q}_i$.

- **Priority Labelling Winning Conditions**:

  - Let $\gamma : V \to P$ be a labelling of the states of the game graph G with priorities in $P = \{1,2,..,|P|\}$.

    For a *global winning condition given by a priority labelling γ*, we say that a joint strategy σ on the game graph G is *winning*, if for any infinite play $v_0 a_1 v_1 a_2 \dots$ that follows σ, the priority sequence $\gamma(v_0)\gamma(v_1)\gamma(v_2)..$ satisfies the parity condition.
  - For each $i \in [n]$, let γ_i be a labelling of the local states of the game graph G with priorities in $P = \{1,2,..,|P|\}$.

    For a *local winning condition given by a priority labellings $\{\gamma_i\}_{i\in[n]}$*, we say that a joint strategy σ on the game graph G is *winning*, if for any agent $i \in [n]$ and for any infinite play $v_0 a_1 v_1 a_2 \dots$ that follows σ, the priority sequence $\gamma_i(v_0^{\downarrow i})\gamma_i(v_1^{\downarrow i})\gamma_i(v_2^{\downarrow i})\dots$ satisfies the parity condition.

Specifying Strategies Given a joint strategy σ of the game graph G, we say that a (V,A)-LTS (S,R,ν,s_ε) *represents* σ, if it satisfies the following: there exists a history $v_0 a_1 v_1 a_2 ...$ that follows σ, if and only if, there exists a path $s_0 a_1 s_1 a_2 ..$ that begins at the start state s_ε and satisfies $\nu(s_i) = v_i$ for all indices i.

Intuitively, for a finite path $s_0 a_1 s_1 a_2 .. s_j$ in such a LTS, the action prescribed by an agent $i \in [n]$ at the corresponding history $\nu(s_0) a_1 \nu(s_1) a_2 .. \nu(s_j)$ is given by $A_i' = \{a^{\downarrow i} \mid sR_a \neq \emptyset\}$. It is easy to see that in the case when the joint strategy σ is deterministic, the set A_i' is a singleton.

Next state the winning strategy problem studied in this thesis. From here on, an *imperfect information game* is denoted by a tuple (G, W), where G is a game graph of a coalition $[n]$ and W is one of the winning conditions stated earlier.

The WINNING STRATEGY PROBLEM for a class $\mathscr{G}$ of imperfect information games, is the following: Is there an algorithm that given an imperfect information game $(G, W) \in \mathscr{G}$, finds a winning joint strategy for the game, if it exists? Here the joint strategy being output is presented as an LTS that represents the joint strategy.

If the joint strategy is required to be deterministic, then we call it the DETERMINISTIC WINNING STRATEGY PROBLEM. We say that the (deterministic) winning strategy problem is *solvable/decidable* for a class $\mathscr{G}$, if the problem can be answered positively.

2.3 Some Observations

Observations on Winning Conditions We begin with a comparison of the expressiveness of the winning conditions mentioned here. Firstly observe that games with winning conditions by ω-automata, for both global and local cases, are inter-reducible with games with winning conditions by priority labellings.

Below we summarize this reduction only for the case of global winning conditions, since the case of local winning condition follows similarly.

- Given a game (G, γ) with priority labelling γ, the desired game $(G, \mathscr{Q})$ where $\mathscr{Q}$ is a parity automaton that has a structure similar to that of the game graph G, with priority labelling on the states given by γ.

- Given a game $(G,\mathscr{Q})$, the desired game (G^-,γ) is obtained as follows: The game graph G^- is obtained from a synchronous product of the game graph G with the transition system underlying the automaton $\mathscr{Q}$, and the priority labelling γ of a product-state of G^- is obtained from the priority label of the automaton-state component of the product-state.

Games with winning conditions given by μ-automata are strictly more general than games with priority labellings; this is not difficult to argue. In view of the above observations, we consider only the case of winning conditions given by priority labellings and μ-automata.

Next we note that in the case of global winning conditions given by μ-automata, the use of 'strong acceptance' by a μ-automaton, as opposed to the more general notion of acceptance, is without loss. This is because for the class of strategy trees, a μ-automaton with the standard acceptance condition can be transformed into one with strong acceptance condition, with an exponential blow-up in the state-space. We show this next.

Proposition 2.3.1. *Consider a game graph G and a μ-automaton $\mathscr{Q}$ that accepts (V,A)-LTS's. Then there exists a μ-automaton $\mathscr{Q}^-$ such that, for any joint strategy σ on G and strategy tree $\mathscr{T}_G^\sigma$ of σ, we have $\mathscr{T}_G^\sigma$ is accepted by $\mathscr{Q}$, if and only if, $\mathscr{T}_G^\sigma$ is strongly accepted by $\mathscr{Q}^-$.*

Proof. This proposition is a consequence of a general property on 'almost-ranked' tree LTS's. A (V,A)-LTS $\mathscr{S} = (S,R,\nu,s_\varepsilon)$ is called an *almost-ranked* tree, if it is a tree, and if for any state s and distinct transitions $(s,a,s_1),(s,a,s_2) \in R$ outgoing at the state s, it is the case that $\nu(s_1) \neq \nu(s_2)$.

We begin by showing that strategy trees are almost-ranked trees. Consider a strategy tree $\mathscr{T}_G^\sigma = (H^\sigma,M^\sigma,\nu,v_\varepsilon)$ of some joint strategy σ. Assume for a contradiction that there exist two distinct transitions $(\tau,a,\tau_1),(\tau,a,\tau_2) \in M^\sigma$ outgoing at the state τ, such that $\nu(\tau_1) = \nu(\tau_2)$. Since $M^\sigma \subseteq M$, it follows from the definition of the transition relation M, that $\tau_1 = \tau a v_1$ and $\tau_2 = \tau a v_2$ for some $v_1, v_2 \in V$. By definition of ν, $\nu(\tau_1) = \mathsf{end}(\tau_1) = v_1$

and $\nu(\tau_2) = \mathsf{end}(\tau_2) = v_2$. Since $\nu(\tau_1) = \nu(\tau_2)$, we obtain that $v_1 = v_2$, a contradiction to the fact that the transitions being considered are distinct.

Next we claim that for a μ-automaton $\mathscr{Q}$, there exists a μ-automaton $\mathscr{Q}^-$ such that, an almost-ranked tree is accepted by $\mathscr{Q}$, if and only if, it is strongly accepted by $\mathscr{Q}^-$. The theorem follows from the claim.

Towards showing this claim, we introduce some terminology. The *ranked tree interpretation* of an almost-ranked tree $(V,A)\mathsf{LTS}$ $\mathscr{S} = (S,R,\nu,s_\varepsilon)$ is given by the $(V,A\times V)$-ranked tree $\mathscr{S}' = (S,R',\nu,v_\varepsilon)$, where the transitions in R' are obtained by replacing the action a on each transition $(s_1,a,s_2) \in R$, with the expanded action $(a,\nu(s_2))$. The fact that the ranked-tree interpretation $\mathscr{S}'$ of the almost-ranked tree $\mathscr{S}$, is a $(V,A\times V)$-ranked tree follows easily, since every outgoing (a,s_2)-transition at a history s_1 goes to the unique history s_2. Also, one can obtain the original tree $\mathscr{S}$ from such a tree $\mathscr{S}'$, by considering each transition $(s_1,(a,\nu(s_2)),s_2)$ in the tree $\mathscr{S}'$ and discarding the component $\nu(s_2)$ from the expanded action $(a,\nu(s_2))$. It is then easy to see that that almost-ranked trees share a bijective correspondence with their ranked tree interpretations.

Now consider a μ-automaton $\mathscr{Q} = (Q,A,V,\delta,\Omega,q_\varepsilon)$. The desired μ-automaton $\mathscr{Q}^-$ claimed above, is constructed in two steps. First we construct a tree automaton $\mathscr{Q}'$ from the μ-automaton $\mathscr{Q}$, that accepts the ranked trees interpretation of the almost-ranked tree accepted by $\mathscr{Q}$. Towards this, construct an alternating tree automaton $\mathscr{Q}_{ATA}$ on $(V,A\times V)$-ranked tree, that ignores the component v in every expanded action $(a,v) \in A\times V$, and operates like the μ-automaton Q. It is easy to see that the language recognized by $\mathscr{Q}$, when restricted to almost-ranked trees, has a bijective correspondence with the ranked trees recognized by $\mathscr{Q}_{ATA}$. By Theorem **??**, there must also exist a tree automaton $\mathscr{Q}' = (Q',A\times V,V,\delta',\Omega',q'_\varepsilon)$, that recognizes the same language as that recognized by $\mathscr{Q}_{ATA}$. Note that, $\mathscr{Q}'$ may have exponentially more states that $\mathscr{Q}_{ATA}$, and since $\mathscr{Q}'$ is a tree automaton, it is also the case that $|f'((a,v'))| = 1$ for all q,v,v',a and $f' \in \delta'(q,v)$.

Next we construct a μ-automaton $\mathscr{Q}^-$ from the automaton $\mathscr{Q}'$, that strongly accepts exactly the set of almost-ranked trees, whose ranked tree interpretation is accepted by $\mathscr{Q}'$. Towards this, let $\mathscr{Q}^- = (Q^-, A, V, \delta^-, \Omega^-, (q'_\varepsilon, \bot))$ such that

- $Q^- = (Q' \times V)$, with $(q'_\varepsilon, v_\varepsilon)$ as the start state,
- for all $q^- \in Q^-$, we have $\Omega^-(q^-) = \Omega'(q^{-\downarrow 1})$, and
- for all $q^- \in Q^-$ and $v \in V$, we have $f^- \in \delta^-(q^-, v)$, if and only if, $q^{-\downarrow 2} = v$, and additionally, there exists $f' \in \delta'(q^{-\downarrow 1}, v)$, such that for all $a \in A$, $f^-(a) = \{(q', v') \in Q' \times V \mid f'((a, v')) = \{q'\}\}$.

We begin with a simple observation about the above automaton.

Lemma 2.3.2. *If $(S, R, \nu, s_\varepsilon)$ is an LTS accepted by $\mathscr{Q}^-$ via a strong acceptance run f_{Q^-}, then $f_{Q^-}(s)^{\downarrow 2} = \nu(s)$ for all $s \in S$.*

Proof. Consider an LTS $(S, R, \nu, s_\varepsilon)$ that is accepted by $\mathscr{Q}^-$ via an acceptance run f_{Q^-}. By definition of f_{Q^-} being accepting, it follows that for every state $s \in S$, $\{(a, f_{Q^-}(sR_a)) \mid a \in A\} \in \delta^-(f_{Q^-}(s), \nu(s))$.

Now for a contradiction, consider a state $s \in S$ such that $f_{Q^-}(s)^{\downarrow 2} \neq \nu(s)$. Firstly observe that for any state $s \in S$, the set $\{(a, f_{Q^-}(sR_a)) \mid a \in A\}$ is non-empty. But if $f_{Q^-}(s)^{\downarrow 2} \neq \nu(s)$, then it follows from the definition of δ^- that $\delta^-(f_{Q^-}(s), \nu(s)) = \emptyset$, a contradiction. □

In order to show that $\mathscr{Q}^-$ is as desired, consider any almost-ranked tree $\mathscr{S} = (S, R, \nu, s_\varepsilon)$ and its ranked tree interpretation $\mathscr{S} = (S, R', \nu, s_\varepsilon)$. To see that $\mathscr{Q}^-$ satisfies desired properties, it suffices to show the following:
$\mathscr{S}'$ is accepted by the tree automaton $\mathscr{Q}'$ via a strong acceptance run, if and only if, $\mathscr{S}$ is strongly accepted by the μ-automaton $\mathscr{Q}^-$ via a strong acceptance run.

By Lemma **??**, it follows that the next claim implies the one above:
For any functions $f_{Q'} : S \to Q'$ and $f_{Q^-} : S \to Q^-$ that satisfy $f_{Q^-}(s) = (f_{Q'}(s), \nu(s))$ for all

$s \in S$, $\mathscr{S}'$ is accepted by the non-deterministic $\mathscr{Q}'$ via the strong acceptance run $f_{Q'}$, if and only if, $\mathscr{S}$ is strongly accepted by the μ-automaton $\mathscr{Q}^-$ via a strong acceptance run f_{Q^-}.

Next, we make three observations, from which the above claim follows immediately.

- Firstly note that for the state s_ε, $f_{Q'}(s_\varepsilon) = q'_\varepsilon$, if and only if, $f_{Q^-}(s_\varepsilon) = (q'_\varepsilon, v_\varepsilon)$. This is an easy consequence of the definition of an acceptance for $\mathscr{Q}'$ and $\mathscr{Q}^-$.

- Secondly note that for any infinite sequence of states $s_1, s_2, ..$ and actions $a_1, a_2, ..$, the sequence $s_1, a_1, s_2, a_2, ..$ is a path in the almost-ranked tree $\mathscr{S}$, if and only if, $s_1, (a_1, \nu(s_2)), s_2, (a_2, \nu(s_3)), ..$ is a path in the ranked tree $\mathscr{S}'$. This is an easy consequence of the definition of the transition relation R' in the ranked tree interpretation $\mathscr{S}'$. Moreover, it follows from the definition of Ω^- and Lemma **??** that, the priority sequence $\Omega'(f_{Q'}(s_1)), \Omega'(f_{Q'}(s_2)), ..$ satisfies the parity condition, if and only if, the priority sequence $\Omega^-(f_{Q^-}(s_1)), \Omega^-(f_{Q^-}(s_2)), ..$ satisfies the parity condition.

- Now consider the functions $f'_\tau : A \times V \to \mathscr{P}(Q')$, $f^-_\tau : A \to \mathscr{P}(Q^-)$, one for each history $\tau \in H^\sigma$, defined as follows:

  - $f'_\tau((a,v)) = \{f_{Q'}(\tau') \mid \tau' \in \tau M'^{\sigma}_{(a,v)}\}$, for each $(a,v) \in A \times V$.

  - $f^-_\tau(a) = \{f_{Q^-}(\tau') \mid \tau' \in \tau M^\sigma_a\}$, for each $a \in A$.

  We claim that $f'_\tau \in \delta'(f_{Q'}(\tau), \nu(\tau))$, if and only if, $f^-_\tau \in \delta^-(f_{Q^-}(\tau), \nu(\tau))$, for every history $\tau \in H^\sigma$.

  Now consider a history $\tau \in H^\sigma$ and the below mentioned equations.

$$
\begin{aligned}
& f^-_\tau(a) \\
=& \{f_{Q^-}(\tau') \mid \tau' \in \tau M^\sigma_a\} && \text{(By definition of } f^-_\tau) \\
=& \{f_{Q^-}(\tau a v) \mid v \in \mathsf{end}(\tau M^\sigma_a)\} && \text{(By definition of } \tau M^\sigma_a) \\
=& \{(f_{Q'}(\tau a v), v) \mid v \in \mathsf{end}(\tau M^\sigma_a)\} && \text{(Since } f_{Q^-}(s) = (f_{Q'}(s), \nu(s))) \\
=& \{(q', v) \mid v \in \mathsf{end}(\tau M^\sigma_a) \text{ and } f_{Q'}(\tau a v) = q'\} && \\
=& \{(q', v) \mid \tau' \in \tau M^\sigma_a \text{ and } f_{Q'}(\tau') = q'\} && \text{(By definition of } \tau M^\sigma_a) \\
=& \{(q', v) \mid f'_\tau((a, v)) = \{q'\}\} && \text{(By definition of } f'_\tau)
\end{aligned}
$$

Now from the definition of δ^- and the above observation that $f^-_\tau(a) = \{(q', v) \mid f'_\tau((a,v)) = \{q'\}\}$, it follows easily that $f'_\tau \in \delta'(f_{Q'}(\tau), \nu(\tau))$, if and only if, $f^-_\tau \in \delta^-(f_{Q^-}(\tau), \nu(\tau))$.

This completes the proof of the proposition. $\square$

Observation on Extended Strategy Trees We now state a characterization of the MaTS underlying an extended strategy tree.

Proposition 2.3.3. *Let $\mathscr{T}_G$ be the game graph specified by a game graph G.*

1. *An induced substructure of the game tree $\mathscr{T}_G$, is the MaTS underlying an extended strategy tree of a joint strategy, if and only if, it is non-terminal, strategic and uniform.*

2. *An induced substructure of the game tree $\mathscr{T}_G$, is the MaTS underlying an extended strategy tree of a deterministic joint strategy, if and only if, it is non-terminal, deterministic, strategic and uniform.*

Proof. We argue both the above items simultaneously. Let the game graph $G = (V, E, v_\varepsilon)$ and let $\mathscr{T}_G = (H, M, \{\approx_i\}_{i \in [n]}, v_\varepsilon)$ be the game graph specified by G.

($\Rightarrow$) For the forward direction, consider the MaTS $\mathscr{E}\mathscr{T}^\sigma = (H^\sigma, M^\sigma, \{\approx^\sigma_i\}_{i\in[n]}, v_\varepsilon)$ extended strategy tree of a joint strategy σ. We need to show that $\mathscr{E}\mathscr{T}^\sigma$ is non-terminal, strategic and uniform, and that when σ is deterministic, then $\mathscr{E}\mathscr{T}^\sigma$ is also deterministic.

Recall that H^σ is the set of histories that follow σ, and that τM_a denotes the a-neighbourhood of τ in $\mathscr{T}_G$. We begin with a simple claim.

Lemma 2.3.4. *For any history $\tau \in H^\sigma$, we have $\tau M_a \cap H^\sigma \neq \emptyset$, if and only if, $a \in \sigma(\tau)$. Moreover, if $a \in \sigma(\tau)$, then $\tau M_a \subseteq H^\sigma$.*

Proof. For the forward direction of the claim, observe that if $\tau M_a \cap H^\sigma \neq \emptyset$, then the definition of τM_a implies that there exists a history $\tau a v \in H^\sigma$, for some $v \in V$. Since every history in H^σ follows σ, it must be the case that a is one of the actions prescribed at τ, that is, $a \in \sigma(\tau)$. This completes the argument for the forward direction. For the reverse direction, if $a \in \sigma(\tau)$, then the a-neighbourhood of τ in M, given by $\tau M_a = \{\tau a v \mid v \in \mathsf{end}(\tau) E_a\}$, are histories that follow σ. Therefore they belong to H^σ, that is $\tau M_a \subseteq H^\sigma$. Moreover since every state in the game graph G has an outgoing transition for every action, the set $\mathsf{end}(\tau) E_a$ is non-empty, and therefore τM_a is non-empty. The desired claims follow. $\square$

- For $\mathscr{E}\mathscr{T}^\sigma$ to be non-terminal, we need to show that every history in H^σ has an outgoing transition. Towards this, observe that the joint strategy σ prescribes an action at every history, that is, for a history $\tau \in H^\sigma$, an action $a \in \sigma(\tau)$ prescribed at τ. Therefore, using Lemma **??**, we have that $\tau M_a \cap H^\sigma \neq \emptyset$. It follows that there exists an outgoing transition at τ in $\mathscr{E}\mathscr{T}^\sigma_G$.

- For $\mathscr{E}\mathscr{T}^\sigma_G$ to be strategic, $\mathscr{E}\mathscr{T}^\sigma_G$ must be an induced substructure of $\mathscr{T}_G$ and the start state v_ε must belongs to $\mathscr{E}\mathscr{T}^\sigma_G$; these are true by definition of $\mathscr{E}\mathscr{T}^\sigma_G$. Additionally, $\mathscr{E}\mathscr{T}^\sigma_G$ must satisfy the following: For any history τ, if $\tau M_a \cap H^\sigma \neq \emptyset$ for some history τ, then $\tau M_a \subseteq H^\sigma$. This follows straightforwardly from our Lemma **??**.

- For $\mathscr{E}\mathscr{T}^\sigma_G$ to be uniform, it must satisfy the following conditions:

  - for any states $\tau_1, \tau_2 \in H^\sigma$ and agent $i \in [n]$, if $\tau_1 \approx^\sigma_i \tau_2$ holds, then $\{a^{\downarrow i} \in A_i \mid \tau_1 M^\sigma_a \neq \emptyset\} = \{a^{\downarrow i} \in A_i \mid \tau_2 M^\sigma_a \neq \emptyset\}$, and

– for any state $\tau \in H^\sigma$, $\{a \in A \mid \tau M^\sigma_a \neq \emptyset\} = \prod_{i\in[n]} \{a^{\downarrow i} \mid \tau M^\sigma_a \neq \emptyset\}$.

Now by definition of a joint strategy, we have that σ satisfies the following:

– for any states $\tau_1, \tau_2 \in H$ and agent $i \in [n]$, if $\tau_1 \approx_i \tau_2$ holds, then $\{a^{\downarrow i} \in A_i \mid a \in \sigma(\tau_1)\} = \{a^{\downarrow i} \in A_i \mid a \in \sigma(\tau_2)\}$, and

– for any state $\tau \in H$, $\{a \in A \mid a \in \sigma(\tau)\} = \prod_{i\in[n]} \{a^{\downarrow i} \mid a \in \sigma(\tau)\}$.

The uniformity of $\mathscr{E}\mathscr{T}^\sigma_G$ follows from this, since $H^\sigma \subseteq H$, and since the predicates $\tau M^\sigma \neq \emptyset$ and $a \in \sigma(\tau)$ are equivalent, due to Lemma **??**.

Now if the joint strategy σ is deterministic, then $\mathscr{E}\mathscr{T}^\sigma_G$ is deterministic. To see this, consider for a contradiction, a history $\tau \in H^\sigma$, two distinct actions a_1, a_2 and transitions $(\tau, a_1, \tau_1), (\tau, a_2, \tau_2) \in M^\sigma$ for some $\tau_1, \tau_2 \in H^\sigma$. Then it follows from the definition of the transition relation M^σ, that $\tau M^\sigma_{a_1} \neq \emptyset$ and $\tau M^\sigma_{a_2} \neq \emptyset$. Moreover since $M^\sigma \subseteq M$, it follows that $\tau M_{a_1} \cap H^\sigma \neq \emptyset$ and $\tau M_{a_2} \cap H^\sigma \neq \emptyset$. By Lemma **??**, this would imply that $a_1, a_2 \in \sigma(\tau)$, whereas for a deterministic joint strategy $\sigma(\tau)$ must be a singleton, a contradiction.

($\Leftarrow$) For the reverse direction, consider a MaTS $\mathscr{H}' = (H', M', \{\approx'_i\}_{i\in[n]}, v_\varepsilon)$ that is an induced substructure of $\mathscr{T}_G$ that is non-terminal, strategic and uniform. We need to show that $\mathscr{H}'$ is the extended strategy tree of some joint strategy.

Towards this, we define a function $\sigma : H \to (2^A \setminus \emptyset)$ as follows:

- If $\tau \in H'$, then $\sigma(\tau) = \{a \in A \mid \tau M'_a \neq \emptyset\}$,
- If $\tau \notin H'$, then for each $i \in [n]$, let $A'_i := \{a^{\downarrow i} \mid \exists \tau' \in H' : \tau \approx_i \tau'$ and $a \in \sigma(\tau)\}$, and let $A' := A'_1 \times A'_2 ... \times A'_n$. Then, $\sigma(\tau) = A'$, if $A' \neq \emptyset$, otherwise $\sigma(\tau) = A$.

To see that the function σ is well-defined, consider a history $\tau \in H'$, and observe that since $\mathscr{H}'$ is non-terminal, there exist some outgoing transition at τ, that is, $\tau M'_a \neq \emptyset$ for some

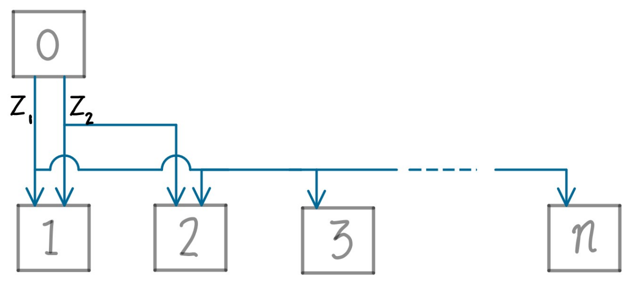
$a \in A$. Therefore by the above definition of σ, $\sigma(\tau)$ is assigned a non-empty set of actions. The function is easily seen to be well-defined at other histories.

We claim firstly that σ is a joint strategy, and secondly that $\mathscr{H}'$ is the extended strategy tree of σ. For the first claim, consider the two defining conditions for being a joint strategy, as given in the forward direction of the proof. Observe that both these conditions are assertions about histories in H. Now in the case when the histories being considered belong to H', both the conditions hold true by virtue of the uniformity condition on the MaTS $\mathscr{H}'$, and the equivalence between the predicates $\tau M^{\sigma} \neq \emptyset$ and $a \in \sigma(\tau)$. In all other cases, the construction of σ for the histories ensures each of the above two conditions. The extra condition in the definition of σ, in the case when $\tau \notin H'$, is defined precisely for this to go through, and this can be verified easily.

For the proof of the second claim, it suffices to show that H' is exactly the set of histories that follow σ. We argue this by an induction on the length of histories. Now for any $k \in \mathbb{N}$, let H'_k denote the set of histories in H' of length k and let H^{σ}_k denote the set of histories of length k that follow σ. The base case $H'_1 = \{v_{\varepsilon}\} = H^{\sigma}_1$ follows easily. Assuming that $H'_k = H^{\sigma}_k$, the argument below shows that $H'_{k+1} = H^{\sigma}_{k+1}$.

$$\begin{aligned}
H'_{k+1} &= \{\tau a v \mid \tau \in H'_k \text{ and } \tau a v \in \tau M'_a\} && \text{(By definition of } H'_{k+1}\text{)}\\
&= \{\tau a v \mid \tau \in H'_k \text{ and } \tau a v \in \tau M_a \cap H'\} && \text{(By definition of } M'\text{)}\\
&= \{\tau a v \mid \tau \in H'_k, \tau M'_a \neq \emptyset \text{ and } \tau a v \in \tau M_a\} && \text{(Since } \mathscr{H}' \text{ is strategic}\\
& && \tau M_a \cap H' \neq \emptyset \text{ implies } \tau M_a \subseteq H'.\text{)}\\
&= \{\tau a v \mid \tau \in H^{\sigma}_k, a \in \sigma(\tau) \text{ and } \tau a v \in \tau M_a\} && \text{(Since } H'_k = H^{\sigma}_k,\\
& && \text{and by definition of } \sigma\text{)}\\
&= H^{\sigma}_{k+1} && \text{(By definition of } H^{\sigma}_{k+1}\text{)}
\end{aligned}$$

Therefore by induction principle, the second claim holds true. This completes the proof of the theorem. □



(a)

Figure 2.3: An architecture on variable-set $\{z_1, z_2\}$.

Here the input variables are $In_0 = \emptyset$, $In_1 = In_2 = \{z_1, z_2\}$ *and* $In_i = \{z_1\}$ *for all* $[n] \setminus \{1, 2\}$, *and the output variables are* $Out_0 = \{z_1, z_2\}$ *and* $Out_i = \emptyset$ *for all* $i \in [n]$.

2.4 The Distributed Synthesis Problem

Towards the description of the distributed synthesis problem, we first informally describe the operation of a distributed program on an architecture, and then construct a game graph that simulates this operation.

An *architecture* on a set of (boolean) *variables* Z is a labelled graph $([n]^+, \alpha)$, where

- $[n]^+ = [n] \cup \{0\}$ is the set of *sub-systems* (or *agents*), with $[n]$ denoting the set of agents in the system excluding the environment agent 0.
- $\alpha \subseteq [n]^+ \times (2^Z \setminus \emptyset) \times [n]$ is such that, $(i, Z', j) \in \alpha$ denotes that agent i has access to write on variables in Z' and agent j has access to read variables in Z'. Additionally, only one agent has write-access for a variable.

For an agent $i \in [n]$, let In_i denote the set of *input* variables that can be read by agent i and let Out_i denote the set of *output* variables that agent i can write on.

A distributed program on an architecture operates in rounds, and can be described as follows: We assume for convenience that before the first round, every variable holds the value 0. At a round, every agent in $i \in [n]$ reads the values of variables in In_i from the previous round, and writes a value for each of its output variables Out_i. The system proceeds by repeating such a round forever.

The game graph G_{Ar} *corresponding* to the architecture $\mathsf{Ar} = ([n]^+, \alpha)$ is defined as the A-TS (V, E, v_ε), where

- $V = \prod_{i \in [n]} V_i$, such that $V_i = \prod_{z \in \mathsf{In}_i} \{[z=0],[z=1]\}$ for each agent i,
- $v_\varepsilon \in V$ such that $v_\varepsilon^{\downarrow i \downarrow z} = [z=0]$ for each agent i and and $z \in \mathsf{In}_i$,
- $A = \prod_{i \in [n]} A_i$, such that $A_i = \prod_{z \in \mathsf{Out}_i} \{[z=0],[z=1]\}$ for each agent i,
- $E \subseteq V \times A \times V$ is such that $(v, a, v') \in E$, if and only if, for any agents $i, j \in [n]$ and variable $z \in \mathsf{Out}_i \cap \mathsf{In}_j$, we have $a^{\downarrow i \downarrow z} = v'^{\downarrow j \downarrow z}$.

Intuitively, A_i denotes the set of actions of agent i and an action $a_i \in A_i$ with $a_i^{\downarrow z} = [z = x]$ denotes that the action a_i assigns a value x to the variable z. The set V_i denotes the set of local states of agent i, and a state $v_i \in V_i$ with $v_i^{\downarrow z} = [z = x]$ denotes that at the local state v_i of agent i, the value of variable z is x. A transition $(v, a, v') \in E$ represents the situation where each agent i assigns a value to the variable $z \in \mathsf{Out}_i$ according to $a^{\downarrow i \downarrow z}$, and each agent i reads the new values of variables $z \in \mathsf{In}_i$ as updated in $v'^{\downarrow i \downarrow z}$. It is easy to see from the definition of transition relation E that at any state v reachable from the start state, the value of any variable $z \in \mathsf{In}_i \cap \mathsf{In}_j$ read by any pair of agents i, j is identical, that is, $v^{\downarrow i \downarrow z} = v^{\downarrow j \downarrow z}$.

The terminology required to state the distributed synthesis problem can now be defined as follows:

- A *distributed program* on an architecture Ar is a *coalition strategy* of G_{Ar}. A *program for an agent* i in the distributed program corresponds to the strategy of the agent i in

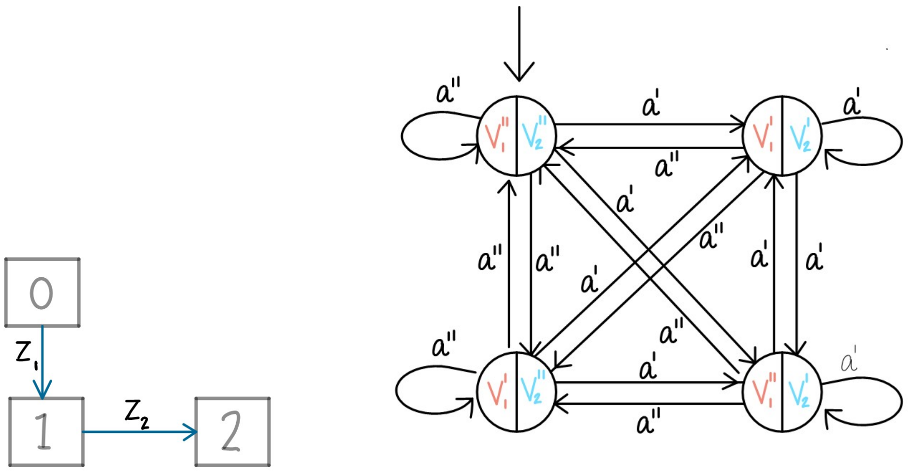


Figure 2.4: An architecture and its corresponding game graph corresponding.

The correspondence is given by the following: $\{a',a''\} = \{[z_2 = 0], [z_2 = 1]\}$, $\{v'_1, v''_1\} = \{[z_1 = 0], [z_1 = 1]\}$ *and* $\{v'_2, v''_2\} = \{[z_2 = 0], [z_2 = 1]\}$.
Here the start state is $([z_1 = 1], [z_2 = 1])$ *instead of the usual* $([z_1 = 0], [z_2 = 0])$

the coalition strategy. We qualify a distributed program or a program for an agent i as *deterministic*, if the strategies being referred to in their respective definitions, are deterministic. Since coalition strategies and joint strategies are equivalent notions, from here on, we will treat a distributed program as a joint strategy.

- The *computation tree* for a distributed program is the strategy tree of the distributed program.

- An *execution specification* (or a *specification*) of a distributed program is simply a winning condition, and we qualify it as *global/local*, if the winning condition is global/local. We say that a distributed program *satisfies* the specification, if the distributed program is winning for the corresponding winning condition

The DISTRIBUTED SYNTHESIS PROBLEM for a class of architectures $\mathscr{A}$ and a class of specifications $\mathscr{S}$, is the following: Does there exist an algorithm, that given an architecture Ar in $\mathscr{A}$ and a specification W in $\mathscr{S}$, returns a distributed program on Ar that satisfies W if it exists?

If the distributed program is required to be deterministic, then we call the above problem DETERMINISTIC DISTRIBUTED SYNTHESIS PROBLEM. We say that the (deterministic) distributed synthesis problem is *solvable* or *decidable* for an architecture class $\mathscr{A}$ and specification class $\mathscr{S}$, if this question can be answered positively.

A (deterministic) distributed synthesis problem may be reduced to a (deterministic) winning strategy problem as follows: an input instance to the (deterministic) distributed synthesis problem, comprised of an architecture Ar and a specification W, maps to the input instance (G_{Ar}, W) of the (deterministic) winning strategy problem. By the terminological correspondence given earlier, it follows that a (deterministic) distributed program on an architecture Ar that satisfies W, is also a (deterministic) winning joint strategy for the game (G_{Ar}, W), and vice-versa.

2.5 Known Results

Distributed Synthesis Problem with Global/Local Specs We state here only the most general results known about the distributed synthesis problem. In the following, we use the terms *global μ-automata specifications* to denote the class of global specifications given by μ-automata, and *local ω-automata specifications* to denote the class of local specifications given by a set ω-automaton, one for each agent in $[n]$.

- **Global Specs**: The state-of-the-art result in the study distributed synthesis problem for global specifications, for both the deterministic and non-deterministic versions, is by Finkbeiner and Schewe in [4] and [25]. We state these results next, and postpone the definition of these architecture classes to Section 2.1.

  Theorem 2.5.1.

  1. *The* deterministic *distributed synthesis problem is solvable for an architecture class $\mathscr{A}$ and global μ-automaton specifications, if and only if, the architectures*

in $\mathscr{A}$ admit a weak-informedness ordering on its agents.

2. *The distributed synthesis problem is solvable for an architecture class $\mathscr{A}$ and global μ-automaton specifications, if and only if, the architectures in $\mathscr{A}$ admit a hierarchy ordering on its agents.*

- **Local Specs**: The distributed synthesis problem for local specifications was characterized for acyclic architectures by Madhusudhan and Thigarajan in [5], by means of 'flanked pipelines'. This result was extended by Fridman and Puchala in [6] to 'flanked pipelines with feedback channels'. The formal definition of these architecture classes can be found in Section 3.1.

  Theorem 2.5.2.

  1. *The* deterministic *distributed synthesis problem is solvable for an acyclic architecture class $\mathscr{A}$ and local ω-automata specifications, if and only if, the architectures in $\mathscr{A}$ are flanked pipelines or their sub-architectures*[2].
  2. *If the architectures in a class $\mathscr{A}$ are flanked pipelines with feedback channels or their sub-architecture, then the deterministic distributed synthesis problem is solvable for the architecture class $\mathscr{A}$ and local ω-automata specifications.*

Game Graphs with Global/Local Winning Conditions After the initial solution of Peterson and Reif [1], there has been little progress in the study winning strategy problems for imperfect information games of the kind studied here. The case of local winning conditions and joint strategies that are not deterministic are yet to be explored from the game point of view.

An exception to this is the approach of *tracking construction* studied in [16]. The construction reduces an imperfect information game (G, γ) specified by a game graph G, with a global winning condition given by an 'observable' priority labellings into a perfect information game $(\mathscr{G}, \gamma')$, where each state of the reduced game corresponds to an 'epistemic

[2]A *sub-architecture* of an architecture is simply an induced subgraph of the architecture.

state' of the system. Intuitively, the epistemic state models the 'information' and 'mutual information' held by agents of the coalition. The main result of [16] shows that winning strategy problem for imperfect information games (G, γ) reduces to a possibly infinite perfect information game $(\mathscr{G}, \gamma')$. Moreover, they show that any such perfect information game $(\mathscr{G}, \gamma')$, with states that correspond to epistemic states, may be reduced further as follows: any two homomorphically equivalent epistemic states in the reduced game graph $\mathscr{G}$, maybe identified as a single epistemic state, without changing the winning status of the game. Consequently, the winning strategy problem is decidable for a class of games, whenever the reduced game is guaranteed to generate only finitely many epistemic states, up to homomorphic equivalence. They show that hierarchical games belong to the class of imperfect information games for which the above transformation yields finitely many epistemic states upto homomorphic equivalence.

2.6 Discussion

Having described the setting of the (deterministic) distributed synthesis problem and its relation to the (deterministic) winning strategy problem, we note that when compared to the standard formulation of the distributed synthesis problem, our formulation here is slightly different in some respects. In the standard formulations, the environment agent is treated as a first-class participant in the distributed system, and the joint actions and states of the system in this case include in them a component for the actions and states of the environment-agent. The reasons for discarding this are two fold: the first is that while we do not record the actions of the environment, the effect of these actions is reflected in the evolution of the states of the system, and therefore discarding environment agent from the analysis is in no way less general; second, the actions and the information available to the environment agent are of no consequence for synthesizing the distributed program, only the effect of these actions need be recorded.

Another point of difference from the setting of distributed synthesis problem is the notion of view_i. In the case of the distributed synthesis problem, the information of an agent at a history does not include the actions executed by the agent, that is, $\mathsf{view}_i(\tau a v)$ is defined as $\mathsf{view}_i(\tau)v^{\downarrow i}$, as opposed to $\mathsf{view}_i(\tau)a^{\downarrow i}v^{\downarrow i}$. This distinction is inconsequential in the case of deterministic joint strategies, since one can uniquely determine the actions executed by an agent along a history, from the sequence of local states visited by the agent. However for the case of non-deterministic joint strategies, this distinction is relevant. We will revisit this issue in Chapter 5. Despite the difference, the solutions studied in this thesis can be modified appropriately for any of these two notions.

An aspect where we will be lax is the question on computational complexity of solvability. The reason for this is that even in positive cases, the algorithms have non-elementary complexity, and therefore are of little practical use. We believe that in view of the current state of research on this problem, the priority for now should be to build better techniques and understanding which distributed synthesis problems are solvable, and this thesis is intended as a step in this direction.

Chapter 3

Games with Recurring Common Knowledge of State

3.1 Introduction

In this chapter, we identify a new condition for the solvability of imperfect information games, that does not rely on the game being hierarchical. The condition, called *recurring common knowledge of the state*, asks that there be infinitely many histories along every play at which the state of the game graph is ‘common knowledge’ to all agents of the agents. The results of this chapter were published in [26],[27].

For the purpose of this chapter, we consider only the case of deterministic joint strategies problem, and therefore refer to them as simply joint strategies. The results in this chapter are the following:

1. The question of whether a game graph satisfies the condition of recurring common knowledge of the state is solvable in NLogSpace.
2. The question of whether there exists a winning joint strategy for a game is NExpTime-complete. Moreover, the winning joint strategy can be synthesized in 2ExpTime.

The solution relies on three key arguments. Firstly, under recurring common knowledge of the state, the intervals where the current state of the game is not common knowledge are bounded uniformly. Secondly, we characterize recurring *common* knowledge in terms of recurring *mutual* knowledge. Finally, we prove that the problem of solving imperfect-information games with recurring common knowledge of state can be reduced to solving parity games with perfect information, at a relatively low cost in terms of complexity.

Preliminaries For the purpose of this chapter, let us fix an imperfect information game (G,γ), where G is a game graph (V,E,v_ε) and $\gamma : V \to P$ is priority labelling that gives the global winning condition, with $P = \{1,2,..,|P|\}$ denoting the set of priorities. For any state $v \in V$, let G_v denote the game graph (V,E,v) obtained by replacing the start state v_ε of the game graph G, by the state v.

We say that two histories τ, τ' in a game graph G are *connected*, if there exists a sequence of histories $\tau_1, \dots, \tau_k$ that begin at τ and end at τ', and a sequence of agents $i_1, \dots, i_{k-1}$ in $[n]$, such that,

$$\tau_1 \approx_{i_1} \tau_2 \approx_{i_2} \cdots \approx_{i_{k-1}} \tau_k.$$

We say that two plays are *connected*, if every history in one play is connected to some history in the other. In the special case when two histories τ and τ' end at the same state v and are connected via a sequence of histories that also end at v, we say that τ and τ' are *twins*. Note that if two histories τ and ρ that end at the same state v are connected or twins, then for every move $(v,a,v') \in E$, the prolongations $\tau a v'$ and $\rho a v'$ are connected or twins respectively.

3.2 Deciding ω-CKS for Game Graphs

CKS, ω-CKS and Knowledge Gaps We say that the agents attain *common knowledge of the state* (CKS) at history τ in the game graph G, if $\mathsf{end}(\tau) = \mathsf{end}(\tau')$ holds for any pair of connected histories τ, τ' in G. We say that a play π allows for *recurring* common knowledge of the state (ω-CKS) if there are infinitely many histories in π at which the agents attain CKS. We say that a game graph G allows for ω-CKS, if every play in G attains ω-CKS.

A *knowledge gap* in a play π is an interval $[\![\ell, t]\!]$ with $t \geq \ell > 0$, such that the agents do not attain CKS at any 'round' in $[\![\ell, t]\!]$ of the play π. Here by a *round k* of the play π, we mean the unique history that is prefix of π and of length k. The *length* of the gap is $t - \ell + 1$. Hence, a play allows for ω-CKS if the length of every knowledge gap in it is finite. The *gap size* (for CKS) of a play π is the least upper bound on the length of knowledge gaps in π. Likewise, the gap size of a game graph is the least upper bound on the gap size of its plays.

Deciding whether a game allows for ω-CKS by checking that the property holds within parts of the game graphs or on individual plays is not easy. In general, histories at which the agents do not attain CKS may be connected to arbitrarily long chains of indistinguishable histories that end at the same state, before reaching one with a different end state to witness the lack of CKS. Fortunately, there is a way around this obstacle. It turns out that in any game graph that allows for ω-CKS, whenever the agents lack common knowledge of the state at some history τ, then there is a history connected to τ that lacks 'mutual knowledge' of the state. This will allow us to characterize games with recurring common knowledge of the state as those where mutual knowledge of the state is attained over and over again, along every play.

Mutual Knowledge of State We say that the agents attain *mutual knowledge of the state* (MKS) at a history τ in a game graph G, if all indistinguishable histories $\rho \approx_i \tau$ end at the same state as τ, for all agents i. A play π allows for recurring mutual knowledge of the state (ω-MKS) if the agents attain MKS at infinitely many histories along π, and a game (graph) G allows for ω-MKS if all plays in G do.

The link between common and mutual knowledge is made by the notion of connected ambiguous histories. A history τ is *ambiguous* if the agents do not attain MKS at τ, that is, if there exists an indistinguishable history $\rho \approx_i \tau$, for some agent i, which ends at a different state. In this case, we refer to ρ as an *ambiguity witness* for τ and we say that ρ, τ is an *ambiguous pair*. Notice that the agents do not attain CKS at a history τ, if and only if, there exists an ambiguous twin of τ.

Our goal is to show that every play in which the agents do not attain ω-CKS, is witnessed by one where they do not attain ω-MKS. Towards this, we first prove that if the agents never attain CKS in a play, there exists a witnessing play in which all histories are ambiguous.

Lemma 3.2.1. *For any game graph, if there exists a play π along which the agents never attain common knowledge of the state (except for the initial state), then there also exists a play π' along which they never attain mutual knowledge of the state. Moreover, the plays π and π' are connected.*

Proof. For an arbitrary game graph G and a play $\pi = v_0 a_1 v_1 \ldots$, we consider the set T_π of all histories τ in G such that every non-trivial prefix history of τ is ambiguous and connected to the history of the same length in π. As the set T_π is closed under prefix histories, we can view it as a finitely branching tree. We wish to show that if the agents do not attain CKS along π, then every history in π is connected to some history in T_π, and therefore T_π contains an infinite play in G along which the agents never attain MKS.

We prove a stronger property, for every history π_ℓ of length $\ell \geq 1$ in π: If the agents do not attain CKS along π_ℓ (except for the trivial history), then for every ambiguous pair $\tau \approx_i \rho$ connected to π_ℓ there exists a pair $\tau' \approx_i \rho'$, such that

1. $\tau' \in T_\pi$ is a twin of τ, and

2. ρ' ends at the same state as ρ.

For the base case with $\ell = 1$, if the agents do not attain CKS at the history $\pi_1 := v_0 a_1 v_1$, then there exist ambiguous histories connected to π_1, and they all belong to T_π, because the only preceding history is trivial. Hence, for any ambiguous pair $\tau \approx_i \rho$, already $\tau' = \tau$ and $\rho' = \rho$ witness the statement.

For the induction step, suppose the statement holds for $\ell \geq 1$ and assume that the agents do not attain CKS up to (and including) the history $\pi_{\ell+1}$ of length $\ell + 1$. In particular, this means that there exist ambiguous histories connected to $\pi_{\ell+1}$; among these, let us pick an ambiguous pair $\tau av \approx_i \rho cw$, with $v \neq w$. Due to perfect recall, we have $\tau \approx_i \rho$.

We distinguish two cases. (1) If τ and ρ end at different states $v' \neq w'$, by induction hypothesis, there exists a twin $\tau' \in T_\pi$ of τ and a history $\rho' \approx_i \tau'$ that ends at w'. On the one hand $\tau' av$ is a twin of τav. On the other hand, by definition of $\approx_i$, we have $\tau' av \approx_i \rho' cw$. Since $\tau' \in T_\pi$ and $v \neq w$, this also implies $\tau' av \in T_\pi$. (2) Otherwise, suppose τ and ρ end at the same state. As the histories are connected to π_ℓ, the agents do not attain CKS at τ. Hence, there exists an ambiguous twin τ' of τ, and by induction hypothesis, we can choose $\tau' \in T_\pi$. On the one hand, $\tau' av$ is a twin of τav. On the other hand, as τ' and ρ end at the same state, so $\tau' cw$ is a valid history in G, and we have $\tau' av \approx_i \tau' cw$. Again, since $\tau' \in T_\pi$, and $v \neq w$ it follows $\tau' av \in T_\pi$. This completes the induction argument.

In conclusion, for a play π in which the agents do not attain CKS at any round, there exist histories in T_π that are connected to arbitrarily long histories of π. As the tree T_π is finitely branching, it follows from König's lemma that it has an infinite path π'. By construction, each non-trivial prefix of π' is an ambiguous history, and it is connected to the history of π of the same length. Hence, π' describes a play connected to π along which the agents never attain mutual knowledge of the state. □

We are now ready to formulate our characterisation result that will be instrumental for deciding the property of ω-CKS on games graphs.

Theorem 3.2.2. *A game allows for recurring common knowledge of the state if, and only* *if, it allows for recurring mutual knowledge of the state.*

Proof. The *only if* direction is trivial: recurring common knowledge of the state implies recurring mutual knowledge of the state.

For the converse, let us consider a game graph G that does not allow for ω-CKS. Then, there exists a play π in which the agents attain CKS at some round ℓ, but not at any later history. Accordingly, in the game graph G_v starting from the state v that is reached in round ℓ of π, there exists a play along which agents never attain CKS, except for the initial state. Then, by Lemma **??**, there exists a play π' in G_v along which the agents never attain MKS. Furthermore, in the play in G that follows π for the first ℓ rounds and, upon reaching v, proceeds like π', the agents do not attain MKS at the infinitely many histories from round ℓ onwards. Hence, the game G does not allow for ω-MKS, which concludes the proof. □

Before turning to algorithmic questions, let us state the following corollary of arguments from the proofs of Lemma **??** and Theorem **??**, which will be useful for bounding the gap size of games graphs.

Corollary 3.2.3. *For any game graph G, if the agents do not attain common knowledge of the state in a play π along a sequence of rounds $\ell+1,\dots,\ell+t$, then there exists a play π' in G that is connected to π and on which the agents do not attain mutual knowledge of the state along the rounds $\ell+1,\dots,\ell+t$.*

Proof. Let G be a game graph and let π be a play with the stated property, for some $\ell, t > 0$. We assume, without loss of generality, that the agents attain CKS at round ℓ and $t+1$ in π. For the game G_v starting at the state v reached in this round, we consider the suffix τ of π from round ℓ onwards, and construct the tree T_τ of ambiguous histories connected to τ, as in the proof of Lemma **??**. The induction argument from the proof then shows that the history of length t in τ is connected to some ambiguous history $\tau' \in T_\tau$. The histories of τ'

from round 1 to t are ambiguous and each of them is connected to the history of the same length in τ. Hence, the play π' that follows π for the first ℓ rounds, then proceeds like τ' for t rounds, and then again follows π, satisfies the required properties: π' is connected to π and the agents do not attain MKS along the rounds $\ell+1,\dots,\ell+t$. $\square$

From the above observations, it follows that for the purposes of deciding whether a game graph admits the property of ω-CKS, it suffices to show that the game graph admits the property of ω-MKS. Now for the purpose of identifying histories of a play that lack MKS, if we consider all the ambiguity witnesses associated with the histories of the play, then the subtree induced in the game unravelling by such ambiguity witnesses can have unbounded width. To handle this, we associate with every play, a structure called a 'fork-pair', that carefully chooses ambiguity witnesses of the histories in the play that lack MKS.

Fork-Pair and Sequences A *fork-pair* for a play π is a prefix-closed set T of histories that contains, for every level $\ell \geq 0$,

(i) the history π_ℓ of π in round ℓ, and

(ii) at most one history $\rho_\ell \neq \pi_\ell$ with $\rho_\ell \approx_i \pi_\ell$, for some agent i.

A fork-pair T is *complete*, if it additionally satisfies, for every level ℓ:

(iii) if π_ℓ is ambiguous, then T contains an ambiguity witness ρ_ℓ of π_ℓ.

We can view fork-pairs as induced subtrees in the unravelling of G that contain π as a central branch and have width at most two, that is, at most two elements on each level. For convenience, we let ρ_ℓ refer to π_ℓ whenever T contains only π_ℓ at level ℓ. In case π_ℓ and ρ_ℓ end at different states, we say that the level ℓ is a *doubleton*, else it is a *singleton*.

Fork pairs for a fixed play π can be represented by ω-words. We say that an ω-word $\tau \in V(AV)^\omega$ is a *fork sequence* for a play π if it starts with $\tau_0 = v_0$ and there exists a fork-pair T for π such that τ_ℓ is the last action-state pair of ρ_ℓ in T, for every $\ell > 0$.

Lemma 3.2.4. *For every play in an arbitrary game there exists a complete fork-pair.*

Proof. It is convenient to extend the notion of ambiguity witness to knowledge gaps in histories. For a history π and an interval $[\![\ell, t]\!]$, we say that a history π' is an ambiguity witness *along the gap* $[\![\ell, t]\!]$ if π and π' have length at least t, and π'_r is an ambiguity witness for π_r, for every round $\ell \leq r \leq t$. Likewise, for a play π, we say that a play π' is an ambiguity witness *from round ℓ onwards* if π'_r is an ambiguity witness for π_r for every $r \geq \ell$.

Now, consider an arbitrary game G and a play π. By induction on the number of rounds ℓ, we construct a finite or infinite sequence of trees T_ℓ that satisfy the fork-tree conditions (i) and (ii) for the first ℓ levels, and, in addition, the following strengthening of the completeness condition (iii) for the last level ℓ:

(iii)* If, for some $t \geq \ell$, there exists a history in G that is an ambiguity witness for π along the gap $[\![\ell, t]\!]$, then there also exists a prolongation history of ρ_ℓ that is such a witness.

In particular, this implies that whenever the history π_ℓ is ambiguous, the level ℓ in T_ℓ is a doubleton.

Each tree $T_{\ell+1}$ is finite and extends its predecessor T_ℓ by one level, except if the sequence ends at some stage $\ell + 1$, in which case $T_{\ell+1}$ extends T_ℓ with the (infinite) prolongation of π_ℓ to π and with a prolongation play ρ of either π_ℓ or ρ_ℓ that is an ambiguity witness for π from round ℓ onwards.

For the base case, we take the tree T_0 consisting only of the initial history v_ε. For the induction step, suppose that a tree T_ℓ with ℓ levels satisfying the conditions (i), (ii), and (iii)* has been constructed. To extend it to $T_{\ell+1}$, we look at the set R of histories τ that prolong either π_ℓ or ρ_ℓ, and are ambiguity witnesses for π along the gap $[\![\ell + 1, t]\!]$ up to the length t of τ. Now we distinguish three cases. (1) If R is empty, we set $\rho_{\ell+1} := \pi_{\ell+1}$, that is, $\ell+1$ is a singleton level. (2) If R is nonempty, but finite, we pick a history $\tau \in R$ of maximal length, and add $\rho_{\ell+1} := \tau_{\ell+1}$ together with $\pi_{\ell+1}$ as a new level to T_ℓ. (3) Finally, if R is infinite, there exists an infinite play τ in G such that all its histories from round ℓ onwards are in R. This follows from König's lemma, since the histories in R form an infinite tree that is finitely branching (indeed, a subtree of the unravelling of G). In this case, we add the histories π_r and $\rho_r := \tau_r$, for all levels $r > \ell$ and terminate the sequence with this infinite tree $T_{\ell+1}$.

In any case, $\rho_{\ell+1}$ is a history in G and is indistinguishable from $\pi_{\ell+1}$ which is also contained on level $\ell+1$. Condition (iii)* holds trivially in case (3), we shall verify that it is also maintained in case (1) and (2).

For case (1) assume, towards a contradiction, that R is empty and there exists a history π' of length $\ell+1$ that is an ambiguity witness for $\pi_{\ell+1}$. If π'_ℓ ends at the same state as π_ℓ, then the last action-state pair $(a'_{\ell+1}v'_{\ell+1})$ of π' yields a prolongation $\tau = \pi_\ell a'_{\ell+1} v'_{\ell+1}$ that should be included in R, a contradiction. Else, if π'_ℓ ends at a different state than π_ℓ, by perfect recall, π'_ℓ is an ambiguity witness for π_ℓ along the gap $[\![\ell, \ell+1]\!]$, which, by induction hypothesis, implies that there also exists such a witness that prolongs ρ_ℓ and is thus contained in R, again in contradiction to our assumption that $R = \emptyset$.

For case (2), consider a history π' of length $t > \ell$ that is an ambiguity witness for π along the gap $[\![\ell+1, t]\!]$. We claim that there also exists a prolongation of $\rho_{\ell+1}$ with this property. There are two situations to distinguish: If π'_ℓ reaches the same state as π_ℓ, then the history π'' that follows π until round ℓ and then continues like π' belongs to R, and is at most as long as the witness τ chosen to construct $\rho_{\ell+1}$. Hence, τ prolongs $\rho_{\ell+1}$ and is an ambiguity witness for π along the gap $[\![\ell+1, t]\!]$. Otherwise, if π'_ℓ reaches a different state than π_ℓ, then, by perfect recall, we have $\pi'_\ell \approx_i \pi_\ell$, for some agent i, and hence π' is already an ambiguity witness for π along the gap $[\![\ell, t]\!]$. By induction hypothesis, there exists an ambiguity witness π'' for π along the gap $[\![\ell, t]\!]$ that prolongs ρ_ℓ. Hence, $\pi'' \in R$ and, as the history $\tau \in R$ chosen to construct $\rho_{\ell+1}$ is of maximal length, τ prolongs $\rho_{\ell+1}$ and is also an ambiguity witness for π along the gap $[\![\ell+1,t]\!]$.

Clearly, each tree T_ℓ constructed along the induction satisfies the conditions of a complete fork-pair and agrees with its successor $T_{\ell+1}$, up to level ℓ. In conclusion, the sequence converges and the infinite tree T obtained at the limit is a complete fork-pair for π. $\square$

In the following we construct, for any arbitrary game, an ω-word automaton with co-Büchi acceptance condition, that takes as input an (infinite) play π in G and guesses in every run a fork sequence for π.

Proposition 3.2.5. *For any game with m states, the set of plays that do not allow for recurring mutual knowledge of the state is recognisable by a nondetermistic co-Büchi automaton with m^2 states.*

Proof. Let us fix an arbitrary game graph G. We construct an ω-word automaton A with co-Büchi acceptance condition that recognizes the set of histories π in G, for which there exists a fork-pair with only finitely many singleton levels. To witness this, the automaton guesses non-deterministically a fork sequence τ for π and accepts if the states at τ_ℓ and π_ℓ are different, for all but finitely many rounds ℓ.

The states of the automaton are pairs of game states from V: the first component keeps track of the input play, the second one is used for guessing the fork sequence τ. The transition function ensures that the two components evolve according to the moves available in the game graph and that the current input symbol yields the same observation as the second component to some agent i.

Concretely, the co-Büchi automaton A is defined over the input alphabet $A \times V$ on the state set $V \times V$ with initial state (v_0, v_0) and transitions from state (u,u') on input (a,v) to state (v,v') whenever $(u,a,v) \in E$ and $v'^{\downarrow i} = v^{\downarrow i}$, for some agent i, and either $(u',a',v') \in E$ or $(u,a',v') \in E$, for some action $a' \in A$ with $a'^{\downarrow i} = a^{\downarrow i}$ for this agent. The set of final states is $Q \setminus \{(v,v) \mid v \in V\}$; the automaton accepts an infinite input word if all states that occur infinitely often in a run are final.

We claim that an input word $\pi \in V(AV)^\omega$ is accepted by A if, and only if, π corresponds to a play in G, and the agents never attain mutual knowledge of the state along π, from some round onwards.

For the *if* direction, consider a play π along which the agents never attain mutual knowledge of the state from some round onwards. By Lemma **??**, there exists a complete fork-pair T for π, in which all but finitely many levels are doubletons. Let τ be the fork sequence associated to T. Then, the sequence $((\pi_\ell, \tau_\ell))_{\ell<\omega}$ describes a run of A on input π in which non-final states (v,v) occur only at the finitely many positions ℓ corresponding to singleton levels in T, thus witnessing that π is accepted.

For the converse, inputs that do not correspond to histories in G are rejected, by construction of A. Furthermore, if an input word π corresponds to a play with infinitely many histories π_ℓ at which the agents attain mutual knowledge of the state, then every run of the automaton visits a non-final state whenever such an input prefix π_ℓ is read. As this occurs infinitely often, the input π is rejected. □

Next we present the algorithm that decides whether a game graph allows for ω-CKS.

Theorem 3.2.6. *The problem of whether a game graph allows for recurring common knowledge of state is* NLOGSPACE*-complete.*

Proof. According to the Theorem **??**, a game graph G allows for recurring common knowledge of the state, if and only if, it allows for recurring mutual knowledge of the state. Our problem thus reduces to checking whether the language recognized by the co-Büchi automaton A constructed for G in Proposition **??** is non-empty. It is well known that the non-emptiness test for co-Büchi automata is in NLOGSPACE (see, for instance, Vardi and Wolper [28]).

Concretely, a nondeterministic procedure can guess a run of A that leads to a cycle included in the set of final states. This requires only pointers to three states of the automaton: two for the current transition and one for storing a state to verify that a cycle is formed. As each state of the automaton is formed by two states of the game, the overall space requirement is logarithmic in the size of the game graph G. Accordingly, the problem of determining whether a game graph allows for common knowledge of state is in NLOGSPACE.

Hardness for NLOGSPACE follows via a straightforward reduction from directed graph acyclicity, shown to be NLOGSPACE-hard by Jones in [29]: Given a directed graph G, we construct a game graph G' for one agent by taking two disjoint copies of G and assigning all non-terminal nodes with the same observation; each terminal node is assigned with a distinct observation and equipped with a self-loop. Finally, we add a fresh initial state to G', with moves to all other states. Clearly, the game graph G' can be constructed using logarithmic space, and the agent has recurring (mutual, common) knowledge of the state in G' if, and only if, the directed graph G is acyclic.

This shows that the problem of determining whether a game graph allows for common knowledge of the state is NLOGSPACE-complete. □

3.3 Solving games with ω-CKS

The key argument in the solution for winning strategy problem for such games is that knowledge gap size between rounds of certainty of state are bounded. Notice that for a play in an arbitrary game, the length of knowledge gaps may be unbounded, even if the play allows for ω-CKS; its gap size is then infinite. Nevertheless, we show that, if a *game* allows for ω-CKS, then there exists a uniform, finite bound on the length of the knowledge gaps in its plays.

As a preliminary step to this, we show that the gap size of such a game graph is finite. Towards this, observe that due to the construction of the game tree $\mathscr{T}_G$, at any two histories

$\tau, \tau' \in H$ at which the agents attains CKS, and end at the same state, the substructures of the game tree $\mathscr{T}_G$ induced by the histories reachable from τ and the histories reachable from τ', are identical. An easy consequence of this observation is that at any two histories τ, $\tau' \in H$ at which the agents attain CKS, with the same end state v, any two prolongations $\tau a \rho, \tau' a' \rho' \in H$ and agent $i \in [n]$, we have $\tau a \rho \approx_i \tau' a' \rho'$, if and only if, $v a \rho \approx_i^v v a' \rho'$, where $\approx_i^v$ denotes the indistinguishability relation of the agent i, in the game tree specified by G_v. The preservation for the case of common knowledge of state follows immediately, and we note this below for further reference.

Lemma 3.3.1. *Let τ be a history in G at which the agents attain CKS, and let v be its last state. Then, the agents attain CKS at a prolongation history $\tau a \rho$ in G, if and only if, the agents attain CKS at history $v a \rho$ in the game G_v.*

Proposition 3.3.2. *If the game graph G allows for recurring common knowledge of the state, then its gap size is finite.*

Proof. Let G allow for ω-CKS. For each state $v \in V$, we construct a tree T_v that may be understood as the unravelling of G from v, up to common knowledge. The nodes of T_v correspond to the histories in G_v that have no strict, non-trivial prefix at which the agents attain CKS. The edges are labelled with joint actions and correspond to moves in G: for any history τ in the domain of T_v at which the agents do not attain CKS, or for $\tau = v$, we have an edge $(\tau, a, \tau a w)$ whenever $(u, a, w) \in E$, for the last state u of τ. The leaves of T_v thus correspond to the histories in G_v at which the agents attain CKS for the first time (not counting the initial history).

Notice that each of the constructed trees is finite branching and all its paths are finite, according to our assumption that all plays allow for ω-CKS. Hence by König's lemma, every tree in the collection $(T_v)_{v \in V}$ is finite. We claim that the maximal height of a tree in this collection is an upper bound for the length of knowledge gaps in the plays of G.

To show this, we construct a game graph G^{CK} over the disjoint union of all unravelling trees T_v, where we identify every leaf history with the root of the tree associated to its last state. Formally, in each tree T_v, we replace every edge (τ, a, ρ), where ρ is a leaf history ending at w, with an edge (τ, a, w) leading to the root of the tree T_w. This induces a natural bijection h between histories of G and G^{CK}, which is also a bisimulation — clearly, the two game graphs have the same infinite unravelling.

The bijection h preserves CKS: By the argument of Lemma **??**, the agents attain CKS at a history τ in G, if, and only if, they attain CKS at the image $h(\tau)$ in G^{CK}. As a consequence it follows that, on the one hand, every history in G^{CK} at which the agents attain CKS ends at the root of some tree T_v, and on the other hand, for every knowledge gap, i.e., every sequence of consecutive histories $\tau_1, \tau_2, \dots, \tau_t$ in G at which the agents do not attain CKS the image $h(\tau_1), h(\tau_2), \dots, h(\tau_t)$ describes a sequence of consecutive histories in G^{CK} that never visit the root of any tree T_v. Hence, the length t of such a sequence is bounded by the maximal length of a path in any of the trees $(T_v)_{v \in V}$. This concludes the proof. □

Next we show that the gap size for games with ω-CKS is bounded, by using the alternative characterization of such games by the 'fork-pair' criterion.

Theorem 3.3.3. *The gap size of any game with m states that allows for recurring common knowledge of the state is bounded by m^2.*

Proof. Consider a game G with m states that allows for ω-CKS. Towards a contradiction, suppose that in G there exists a play with gap size greater than m^2, that is, the agents do not attain CKS along a sequence of consecutive rounds $r, \dots, r + m^2$, for some r. Due to Corollary **??**, there also exists a play π in G such that the agents do not attain MKS in π along these rounds. Let T be a complete fork-pair for π, according to Lemma **??**, and let τ be the associated fork sequence.

As G allows for ω-MKS, the automaton A constructed in Proposition **??** recognizes the empty language, in particular it rejects the run on π described by (π, τ). But A has at most m^2 states, so there must be a cycle in the transition graph that is visited by this run, say from position $\ell \geq r$ to $t \leq r+m^2$. Along the interval $[\![\ell,t]\!]$, the agents do not attain MKS in π, therefore the corresponding levels in the fork-pair T are doubletons, and the states on the cycle visited in the run (π,τ) from position ℓ to t are final.

Consider now the sequences π', and τ' that follow π and τ, respectively, until position t and then loop from ℓ to t forever. Then, the pair (π',τ') describes a run in A that eventually cycles through final states, hence, the input π' is accepted. But this means that π' is a play in G that does not allow for ω-MKS, in contradiction to our assumption that all plays in G allow for ω-CKS. □

We observe that the quadratic bound on the gap size is tight. Consider, for instance, a number $m \in \mathbb{N}$ and the game graph G_m for two agents as depicted in Figure **??**, where one of the agents, say 1 has two local states, denoted by black and white, and the other agent has a different local state for each state of the game graph, which we do not denote in the figure. At the start state, the environment can either move up, into the cycle with $m-1$ white states followed by a black one, or down, to the path consisting of m white states with self-loops, each followed by a black state, except for the last one which leads to the black state on the cycle. Consider the play π where the environment moves into the cycle and stays there forever. Along π, every non-trivial history up to round m^2 is indistinguishable from the one where environment moves initially down to the path and loops on each white state precisely $m-1$ times. For the first m^2 rounds in π, the agent 1 does therefore not know the current state, which means that the gap size of the game is at least m^2. On the other hand, notice that all histories that are distinguishable from π are non-ambiguous, and that from round m^2+1 onwards, any history that is indistinguishable from π leads to the same state as π itself. Accordingly the game graph G_m with $3m$ states allows for ω-CKS and its gap size is m^2.

We are now ready to establish complexity bounds for the basic algorithmic questions on games with recurring common knowledge of the state. At the end of the section, we explain how the results apply to observable priority labellings.

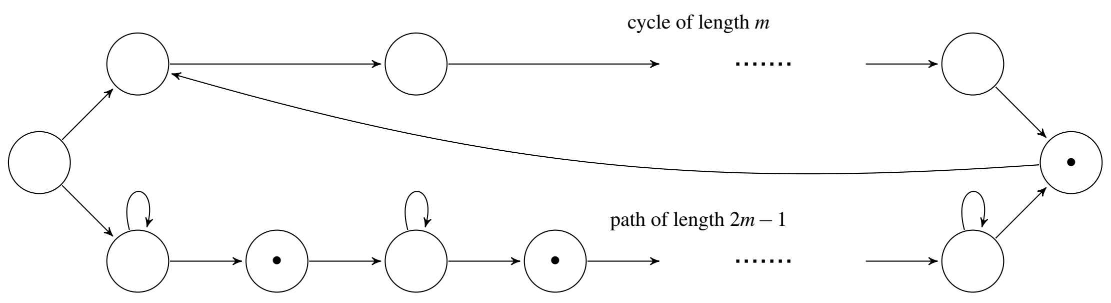


Figure 3.1: A game graph with 3m states and gap size m^2

Theorem 3.3.4. *For games that allow for recurring common knowledge of the state, with priority labellings,*

1. *the problem of deciding whether there exists a winning joint strategy is* NEXPTIME-*complete;*
2. *if winning joint strategies exist, then they can be synthesized in* 2-EXPTIME.

The lower bound for the decision problem (**??**) follows from NEXPTIME-hardness of the corresponding problem for two-agent reachability or safety games of finite horizon. These are games where the underlying graph is acyclic except for having self-loops at observable sinks — hence, the simplest examples of games that allow for CKS. The original proof, due to Azhar, Peterson, and Reif [30, Section 5], is by reduction from the time-bounded halting problem via a variant of QBF with dependency quantifiers.

For the upper bound and the strategy-construction procedure, we introduce an auxiliary representation of the game which retains the histories at which agents attain CKS and is only exponential in the size of the input game graph.

The abridged game The *abridged* game $(\widehat{G}, \widehat{\gamma})$ of (G, γ) is a game with perfect information for one agent against environment. Intuitively, $(\widehat{G}, \widehat{\gamma})$ is obtained by contracting knowledge gaps and recording only the most significant priority seen between two consecutive histories where the agents attain CKS.

Concretely, the states of the abridged game graph $\widehat{G}$ are pairs (v,p) of states $v \in V$ and priorities $p \in P$; for convenience, we also include a sink state $\ominus$, from which there are no outgoing moves. We shall refer to the states of $\widehat{G}$ as *positions*, to avoid confusion with the ones of (G,γ). The initial position (v_ε, p_{max}) corresponds to the initial state of (G,γ) labelled with the most significant priority (that is, the smallest priority) p_{max} in P. The set of actions consists of all non-empty subsets $U \subseteq V \times P$ of positions.

To define the moves, we look at the unravelling G^{CK} up to common knowledge of the game graph G, as constructed in the proof of Proposition **??**. Recall that G^{CK} is built from a disjoint collection of trees $(T_v)_{v\in V}$, which are then connected by identifying all leaves with the corresponding roots. For every state $v \in V$ and any joint strategy t over the tree component T_v of G^{CK}, we define the set $\mathrm{outcome}_v(t)$ of pairs $(u,p') \in V \times P$, for which there exists a history τ in T_v that follows t, such that τ ends at u, and the most significant priority that occurs along τ is p'. Now, the set of available moves is defined as follows. For an action $U \subseteq V \times P$ there are moves from a position (v,p) to every position $(u,p') \in U$ if there exists a joint strategy t in T_v with $\mathrm{outcome}_v(t) = U$. Otherwise, the action leads to the $\ominus$-sink. Notice that the moves depend only on the first component of the position, that is, on the state and not on the priority.

At last, we define a parity condition on (G,γ), by assigning to every position $(v,p) \in V \times P$ the priority p.

The plays of G and $\widehat{G}$ are related via their summaries. Intuitively, this is the sequence of states reached when the agents attain CKS in a play, together with the most significant priority seen along the last knowledge gap. More precisely, for a play $\pi = v_0\, a_1 v_1 \ldots$ in G, we look at the subsequence of rounds $t_0, t_1, t_2, \ldots$ such that, for all $\ell \geq 0$, the agents attain CKS at round t_ℓ in π, but not at any round t in between $t_\ell < t < t_{\ell+1}$. Next, we associate to each index $\ell > 0$, the most significant priority that occurred in the gap between t_ℓ and $t_{\ell+1}$, setting $p_{\ell+1} := \min\{\gamma(v_t) \ : \ t_\ell < t \leq t_{\ell+1}\}$. Now, the *summary* of π is the sequence $[\pi] := v_0, (v_{t_1}, p_1), (v_{t_2}, p_2) \ldots$ Notice that for every play π in G, the summary $[\pi]$

corresponds to a sequence of states in $\widehat{G}$, which is infinite, since we assume that π allows for ω-CKS.

The notion of summary is defined analogously for histories, and it also applies to plays $\hat{\pi}$ in $\widehat{G}$. Indeed, $[\hat{\pi}]$ is obtained simply by dropping the actions in the sequence $\hat{\pi}$. We say that a play π in G *matches* a play $\hat{\pi}$ in $\widehat{G}$ if they have the same summary: $[\hat{\pi}] = [\pi]$.

The next lemma shows that the winning or losing status is preserved among matching plays.

Lemma 3.3.5. *If a play π of (G, γ) matches a play $\hat{\pi}$ of $(\widehat{G}, \widehat{\gamma})$, then π is winning if, and only if, $\hat{\pi}$ is winning.*

Proof. Let p be the least priority that appears infinitely often in π. As each knowledge gap in π is finite, p appears in infinitely many knowledge gaps in π, hence it is recorded infinitely often in the summary $[\pi]$. Conversely, all priorities that appear infinitely often in the summary $[\pi]$, also appear infinitely often in π, so p is minimal among them. In conclusion, the least priority appearing infinitely often in the summaries $[\pi] = [\hat{\pi}]$ is the same as in the plays π and $\hat{\pi}$. $\square$

Reduction to Games with Perfect Information To use results from the standard literature on parity games, it is convenient to view the abridged game $(\widehat{G}, \widehat{\gamma})$ formally as a turn-based game between two agents, *Coordinator* and *Nature*. In contrast to before, we shall hence regard Nature as an actual agent with proper positions, moves, and strategies.

Towards this, we view the game graph $\widehat{G}$ as a bipartite graph, with one partition $V \times P$ controlled by Coordinator, and a second one formed by the non-empty subsets of $V \times P$ denoting the actions of the abridged game, controlled by Nature. The initial position (v_ε, p_{max}) is unchanged. Coordinator can move from every position $(v, p) \in V \times P$ to a position $U \subseteq V \times P$, if $U = \text{outcome}_v(t)$ for some joint strategy t on T_v, whereas Nature can move from every position $U \subseteq V \times P$ to any element $(u, p') \in U$. The new positions from $U \subseteq V \times P$ receive the least significant priority in P, whereas position $(v, p) \in V \times P$ have priority p, as before. The notion of strategy for Coordinator is that of a function that maps histories ending in a position controller by the coordinator, to a successor of this position. The notion of strategy for Nature is similarly defined.

A fundamental result about parity games is that they enjoy *positional determinacy*. A strategy is positional if the choice prescribed at a history π depends only on the last position in π. The following theorem was proved by Emerson and Jutla [31].

Theorem 3.3.6 ([31])**.** *For every parity game with perfect information, one of the two agents has a positional winning strategy.*

For our setting, positional determinacy means that in the abridged game $(\widehat{G}, \widehat{\gamma})$, either Coordinator or Nature has a winning strategy defined on the set of positions. This yields witnesses of manageable size for determining which agent wins the abridged game.

In the following, we argue that positional strategies for the abridged game $(\widehat{G}, \widehat{\gamma})$ can be translated effectively into strategies on (G, γ), such that the resulting plays match in the sense of Lemma **??**.

Proposition 3.3.7. *Let (G, γ) be a game that allows for ω-CKS, and let $\widehat{G}$ be the abridged game.*

1. *For every positional Coordinator strategy $\hat{s}$ in $(\widehat{G}, \widehat{\gamma})$, we can effectively construct joint strategy s for the game (G, γ) such that, for every play π that follows s, there exists a matching play $\hat{\pi}$ that follows $\hat{s}$.*

2. *For every positional Nature strategy $\hat{r}$ in $(\widehat{G}, \widehat{\gamma})$, and every joint strategy s for the game (G, γ), there exists a play π in (G, γ) that follows s with a matching play $\hat{\pi}$ that follows $\hat{r}$.*

Proof. 1. Let $\hat{s} : V \times P \to 2^{V \times P}$ be a positional strategy for Coordinator in the abridged game $(\widehat{G}, \widehat{\gamma})$. We construct a joint strategy s for the unravelling $(G, \gamma)^{\mathrm{CK}}$ of (G, γ) up to common knowledge. As the two game graphs have the same unravelling, s is also a strategy for $\mathscr{G}$.

We can assume that the strategy $\hat{s}$ prescribes for every state $v \in V$ the same choice at all positions (v, p) independently of the priority. This is without loss of generality: Recall that all positions (v, p) in $(\widehat{G}, \widehat{\gamma})$ have the same set of successors U. If we add a fresh position z_v, of least significant priority, from which Coordinator can move to every position in U, and replace the outgoing moves from each position (v, p) with a move to z_v, the game remains essentially unchanged. Whenever Coordinator has a winning strategy for the original game, he has one for the modified game. Then, due to positional determinacy, he also has a positional winning strategy and its choice at the new position z_v can be transferred as a uniform choice to all positions (v, p) in the original game, still yielding a winning joint strategy.

To transfer the given strategy from $(\widehat{G}, \widehat{\gamma})$ to $(G, \gamma)^{\text{CK}}$, we consider for each state $v \in V$ the tree component T_v of $(G, \gamma)^{\text{CK}}$ separately. For an arbitrarily chosen priority p, we look at the set $U := \hat{s}(v, p)$ and pick a joint strategy t_v on T_v with $\text{outcome}_v(t_v) = U$. Now, for every history π that ends in T_v, we take the suffix π_v contained in T_v, that is, we forget the prefix history up to entering the tree, and set $s(\pi) = \{t_v(\pi_v)\}$. This is a valid joint strategyby Lemma **??**.

With s constructed this way, every play π in G that follows s has the same summary $[\pi] = v_0\,(v_1, p_1)(v_2, p_2)\dots$ as the play $\hat{\pi} = v_0\,a_1(v_1, p_1)\,a_2(v_2, p_2)\dots$ in $\widehat{G}$ with actions $a_\ell = \hat{s}(v_\ell, p_\ell)$. Hence, $\hat{\pi}$ follows $\hat{s}$ and matches π, as required.

2. For the converse, let $\hat{r} : 2^{V \times P} \to V \times P$ be a positional strategy for Nature in $\widehat{G}$ and let s be an arbitrary strategy for Coordinator in G. We construct a pair of plays π in G, and $\hat{\pi}$ in $(\widehat{G}, \widehat{\gamma})$ with the desired properties.

The construction is by induction on the number of knowledge gaps in π: For every ℓ, we construct a history π_ℓ in G with ℓ knowledge gaps that follows s and ends at some state v, where the agents attain CKS. At the same time, we construct a matching history $\hat{\pi}_\ell$ that follows $\hat{r}$ and ends at a position (v,p) in $\widehat{G}$, associated with the same state v.

For the base case, both histories π_0 and $\hat{\pi}_0$ are set to v_0. For the induction step, suppose that the two histories π_ℓ and $\hat{\pi}_\ell$ satisfy the hypothesis, and that they end at state v and position (v,p), respectively. We construct a prolongation $\pi_{\ell+1}$ that follows s over the $\ell+1$st knowledge gap and matches a one-round prolongation $\hat{\pi}_{\ell+1}$ of $\hat{\pi}_\ell$. Towards this, we consider the strategy t_v induced by s in the set of histories $\pi_\ell T_v$, that is, the prolongations of π_ℓ into the tree component T_v of G^{CK}. For $U := \text{outcome}_v(t_v)$ and $(u,p') := \hat{r}(U)$, there exists a history τ in T_v that ends at u and has p' as most significant priority after the initial state v. Now, we update $\pi_{\ell+1} := \pi_\ell \tau$ and $\hat{\pi}_{\ell+1} := \hat{\pi}_\ell U(u,d)$. This way, $\pi_{\ell+1}$ follows s and the agents attain CKS, and $\hat{\pi}_{\ell+1}$ follows $\hat{r}$. Moreover, the two plays have the same summary, and $\hat{\pi}_{\ell+1}$ ends at a position corresponding to the last state of $\pi_{\ell+1}$.

For the infinite plays π and $\hat{\pi}$ obtained at the limit, we have: π follows s and matches $\hat{\pi}$ which follows $\hat{r}$, as required.

□

The correspondence between strategies in the abridged game and in the original game results in the following conclusion.

Proposition 3.3.8. *Let (G,γ) be an imperfect-information game that allows for recurring common knowledge of the state, such that G has m states and γ maps states in G to the priorities in $\{1,2,..,d\}$.*

1. *There exists a joint winning strategy for (G,γ) if, and only if, Coordinator has a positional winning strategy for the abridged game $(\widehat{G},\widehat{\gamma})$, that is a perfect-information parity game with $md+2^{md}$ positions and d priorities.*

2. *If there exists a joint winning strategy in (G,γ), then there exists a winning profile of finite-state strategies with $2^{\mathcal{O}(m^2\log m)}$ states.*

Proof. 1. If Coordinator has a positional winning strategy $\hat{s}$ in $(\widehat{G}, \widehat{\gamma})$, then the corresponding profile s according to Proposition **??**(i) is winning in (G, γ), because every play π that follows s has a matching play in (G, γ) that follows $\hat{s}$ and is hence winning, which implies that π is also winning, by Lemma **??**.

Conversely, assume that there exists a joint winning strategy s in G. By Proposition **??**(ii), for any arbitrary positional strategy $\hat{r}$ of Nature, there exists a play $\hat{\pi}$ that follows $\hat{r}$ and matches some play π in G which follows s and thus wins. Hence, $\hat{\pi}$ is also winning for Coordinator, by Lemma **??** which means that $\hat{r}$ in not winning for Nature. By positional determinacy, it follows that Coordinator has a positional winning strategy in $\widehat{G}$.

The state space of the abridged game is $V \times P \cup 2^{V \times P}$, it has $md + 2^{md}$ positions; the number d of priorities is as in (G, γ).

2. Let $\hat{s} : V \times P \to 2^{V \times P}$ be a winning strategy for Coordinator in the abridged game $(\widehat{G}, \widehat{\gamma})$. As in the proof of Proposition **??**(i), we assume, without loss of generality, that the strategy prescribes the same move $U = \hat{s}(v, p)$, at all positions corresponding to v, independently of the priority p; for each of the m states $v \in V$, the move $\hat{s}(v, p)$ is translated into an imperfect-information strategy t_v on the tree component T_v of G^{CK}. We use these local strategies to construct a joint winning strategy s for the grand agents in G as follows.

For each agent i, the component strategy s^i is implemented by a procedure that maintains, along the infinite sequence of states, a record (v, ρ^i) of the last state v about which the agents attained common knowledge and the local-state history ρ^i along the subsequent knowledge gap. Initially v is set to v_0 and ρ^i to $v_0^{\downarrow i}$. In each step, the procedure returns the action $a^i := t_v(\rho)^{\downarrow i}$, inputs the next local state v'^i, and repeats with $\rho^i a^i v'^i$ as a new value for ρ^i, unless this corresponds to a history in G_v at which the agents attain common knowledge of the current state v'. In that case, the root v is replaced with v' and the new value of ρ^i becomes $v'^{\downarrow i}$. Each local strategy t_v^i can be represented by a (tree shaped) automaton that outputs actions in response to observation sequences along knowledge gaps — of length at most m^2, by Theorem **??**. Since there are no more observations than game states, m^{m^2} automaton states are sufficent to store these responses, as well as the set of histories at which the agents attain CKS. Globally, the strategy s^i of each agent i combines m local strategies t^i_v. Hence, we need at most $m \cdot m^{m^2} = 2^{\mathcal{O}(m^2 \log m)}$ many states to represent each component of the profile s by a strategy automaton. $\square$

Next we discuss the complexity of determining the existence of a winning strategy. According to Proposition **??**, this can be done guessing the abridged game $(\widehat{G}, \widehat{\gamma})$ and determining whether the Coordinator has a winning strategy in the obtained parity game with perfect information. The complexity is dominated by the verification of the transition relations between Coordinator positions (v, p) and Nature positions $U \subseteq V \times C$, which involves guessing a witnessing strategy profile t_v over the tree T_v such that $\text{outcome}_v(t_v) = U$. As we pointed out in the proof of Proposition **??**, for a game (G, γ) of size m, such a strategy t_v can be represented by a collection of n trees of size $2^{\mathcal{O}(m^2 \log m)}$, one for every agent. Once the local strategy trees t^i are guessed, the verification that $\text{outcome}_v(t) = U$ is done in time linear in their size. Given the abridged game, a winning strategy for Coordinator can be guessed and verified in non-deterministic linear time with respect to the size $md + 2^{md}$ of $(\widehat{G}, \widehat{\gamma})$ where d is the number of priorites. Overall, the procedure runs in $\text{NTIME}(2^{\mathcal{O}(m^2 \log m)})$, that is, non-deterministic exponential time.

With a deterministic procedure, the abridged game can be constructed by exhaustive search over witnessing strategies over the component trees in G^{CK} in time $2^{2^{\mathcal{O}(m^2 \log m)}}$. Once this is done, winning strategies for the obtained parity game $(\widehat{G}, \widehat{\gamma})$ can be constructed in time $\mathcal{O}(2^{md^2})$ using the basic iterative algorithm presented by Zielonka in [32]. This concludes the proof of Theorem **??**.

3.4 Discussion

Known solvable classes from the distributed-systems literature rely on an ordering on the information available to agents and then use this ordering to decompose the system into parts that are similar to one-player imperfect information games against the environment. Our approach is orthogonal to this idea, and instead of restricting the order of information, our solvability condition requires that players attain common knowledge of the game state infinitely often along every play. The technical manifestation of this condition allows us to decompose the game tree into a sequence of time slices that can be solved independently. We believe that this may be viewed as a situation where, the imperfect information is present only as a temporary perturbation of a perfect information. It remains to be seen whether other such small perturbations in the information structure, for instance, due to communication delays or incomplete but perfect information, still retain the solvability status of the game.

Chapter 4

The Retraction Approach

4.1 Introduction

In this chapter, we introduce the building blocks of a different approach to the winning strategy problem, called the *retraction approach*. Like in the case of tracking construction [16], the retraction approach relies on explicitly modelling the 'information' possessed by agents, and aspires to explain the decidability results for imperfect information games as being due to the presence of certain structural patterns in the indistinguishability relation of the game.

We begin with an intuitive account of the methodology underlying the retraction approach. In the study of algorithms, a typical approach to solving a search problem is to first design a class of structures that is rich enough to represent its solutions, and then search through them until a structure corresponding to a solution is found. We call such a structure, the one that corresponds to a solution, as a *witness*. A requirement for performing such a search is that the search space must be recursively enumerable, and additionally, it should be easy to verify if a finite structure being considered is indeed a witness. As an example of this methodology, consider the satisfiability problem of boolean formulae. The 'brute-force' algorithm for determining the satisfiability of a formula, uses the class of 'valuation functions' (an assignment of variables to valuations) as the class of structures representing the solution. The algorithm for satisfiability proceeds by searching for a valuation function that represents a solution. Moreover, it limits its search space to 'bounded' valuation functions, that only consider valuations of variables that appear in the input formula. The retraction approach solves the winning strategy problem in a similar manner. As a general remark, we note that such a methodology is also 'complete' for the class of decidable problems, in the following sense: For a problem with an algorithm that runs on an input x in time at most $f(|x|)$, the witness is given by the configuration space of the Turing machine covered by the run of the algorithm on the input x: it is of size at most $f(|x|) \times f(|x|)$.

For reference, we state below the main steps of this methodology with respect to the winning strategy problem.

1. First, find a class of structures rich enough to adequately represent winning joint strategies of imperfect information games.
2. Second, restrict the search space to some sub-class of bounded size structures.

Here we use MaLTS's as witnesses to winning joint strategies of a game, and define them via a set of axioms on the the class of MaLTS's. The method used here to restrict the search space of witnesses is by an operation on witnesses, called a *retraction*, that gives some generic conditions to transform a witness into possibly smaller witnesses.

Note that due to undecidability of the winning strategy problem for imperfect information games, there is no hope of finding a retraction operation that uniformly transforms all the witnesses of a game into bounded size witnesses. Our intention for this approach is quite different. Put simply, we hope to identify classes of games for which a certain class of witnesses of the game called 'canonical witnesses', admit a retraction to a bounded size witness. We will see in later chapters that many previous results can be explained from this perspective. In the remaining part of this chapter, we consider variants of imperfect information games and define the notion of witness and retraction for each of them.

4.2 Witnesses and Retractions for Global Specs.

Let us fix an imperfect information game $(G, \mathscr{Q})$, where $G = (V, E, v_\varepsilon)$ is a game graph and $\mathscr{Q}$ is a global winning condition given by the μ-automaton $(Q, A, V, \delta, \Omega, q_\varepsilon)$. Let $\mathscr{T}_G = (H, M, \{\approx_i\}_{i\in[n]}, v_\varepsilon)$ denote the game tree associated with the game graph G. Recall that a joint strategy σ is said to satisfy the winning condition specified by $\mathscr{Q}$ only if the strategy tree of σ is strongly accepted by $\mathscr{Q}$.

Witnesses A $(\{V, Q\}, \{A_i\}_{i\in[n]})$-MaLTS $\mathscr{W} = (S, R, \{\sim_i\}_{i\in[n]}, \{\ell_V, \ell_Q\}, s_\varepsilon)$ is called a *witness* for the imperfect information game $(G, \mathscr{Q})$, if it satisfies the following:

S.1. (1) $\forall s_1, s_2 \in S, \forall i \in [n]:$ if $s_1 \sim_i s_2$, then (Uniformity-1)

$\{a^{\downarrow i} \in A_i \mid s_1 R_a \neq \emptyset\} = \{a^{\downarrow i} \in A_i \mid s_2 R_a \neq \emptyset\}$,

and

(2) $\forall s \in S : \{a \in A \mid sR_a \neq \emptyset\} = \prod_{i\in[n]} \{a^{\downarrow i} \mid sR_a \neq \emptyset\}$ (Uniformity-2)

S.2 $\forall s \in S, \exists a \in A : sR_a \neq \emptyset$. (Non-Termination)

V.1. $\ell_V(s_\varepsilon) = v_\varepsilon$, and $\forall s \in S, \forall a \in A:$ if $sR_a \neq \emptyset$, then (Strategy-Simulation)

$\ell_V(sR_a) = \ell_V(s)E_a$ and $|sR_a| = |\ell_V(s)E_a|$.

V.2 $\forall i \in [n], \forall (s_1, a, s_2), (s'_1, a', s'_2) \in R:$ if $s_1 \sim_i s'_1$, then, (Indistinguishability)

$s_2 \sim_i s'_2$, if and only if, $\ell_V(s_2)^{\downarrow i} = \ell_V(s'_2)^{\downarrow i}$ and $a^{\downarrow i} = a'^{\downarrow i}$.

Q.1. $\ell_Q(s_\varepsilon) = q_\varepsilon$, and ($\mathscr{Q}$-Simulation)

$\forall s \in S : \{(a, \ell_Q(sR_a)) \mid a \in A\} \in \delta(\ell_Q(s), \ell_V(s))$.

Q.2. for every path $s_0 a_1 s_1 a_2 ..$ in the witness, the sequence ($\mathscr{Q}$-Parity)

$\Omega(f_Q(s_0)), \Omega(f_Q(s_1)), ..$ satisfies the parity condition.

Intuitively, the labelling ℓ_V ensures that the witness represents a strategy tree corresponding to some joint strategy; the conditions **S.1**, **S.2**, **V.1**, **V.2** together ensure that the MaTS underlying the witness corresponds to the MaTS underlying the extended strategy tree of some joint strategy. The labelling ℓ_Q ensures that the strategy tree of this joint strategy, satisfies the winning condition $\mathscr{Q}$; the conditions **Q.1**, **Q.2** ensure this.

We begin by showing a correspondence between the class of winning (deterministic) joint strategies and the class of (deterministic) witnesses.

Theorem 4.2.1 (Witness Theorem). *For any imperfect information game $(G,\mathscr{Q})$, there exists a winning (deterministic) joint strategy for $(G,\mathscr{Q})$, if and only if, there exists a (deterministic) witness for $(G,\mathscr{Q})$.*

Proof. ($\Rightarrow$) We begin with the forward direction of the theorem. Towards this, assume that there exists a winning joint strategy σ for the game $(G,\mathscr{Q})$. Let H^σ denote the set of histories that follow σ, let $\mathscr{E}\mathscr{T}_G^\sigma = (H^\sigma, M^\sigma, \{\approx_i^\sigma\}_{i\in[n]}, \nu, \nu_\varepsilon)$ denote the extended strategy tree of σ, and let $\mathscr{T}_G^\sigma = (H^\sigma, M^\sigma, \nu, \nu_\varepsilon)$ be the strategy tree of σ. Recall that $\nu(\tau) = \mathsf{end}(\tau)$ for all $\tau \in H^\sigma$. By definition of a winning joint strategy, the LTS $\mathscr{T}_G^\sigma$ is strongly accepted by the μ-automaton $\mathscr{Q}$ via some strong acceptance run, say f_Q.

We claim that the structure $\mathscr{W} = (H^\sigma, M^\sigma, \{\approx_i^\sigma\}_{i\in[n]}, \{\ell_V, \ell_Q\}, \nu_\varepsilon)$, with $\ell_V(\tau) = \mathsf{end}(\tau)$ and $\ell_Q(\tau) = f_Q(\tau)$ for all $\tau \in H^\sigma$, is a witness for the game $(G,\mathscr{Q})$. We call such a witness the **canonical witness** associated with the winning joint strategy σ.

Next, we argue that $\mathscr{W}$ satisfies each of the conditions for being a witness, and is therefore a witness.

- **S.1**,**S.2**: The fact that $\mathscr{W}$ satisfies the conditions **S.1**,**S.2**, is a consequence of two facts: firstly that the MaTS underlying the witness $\mathscr{W}$ is identical to the MaTS underlying the extended strategy tree $\mathscr{E}\mathscr{T}_G^\sigma$; and secondly, by Proposition **??**, such a MaTS underlying an extended strategy tree is uniform and non-terminal. The conditions **S.1**,**S.2** state exactly this.

- **V.1**: The first part of this condition requires that $\ell_V(\nu_\varepsilon) = \nu_\varepsilon$, and this follows immediately from the definition of ℓ_V. For the second part, consider a history $\tau \in H^\sigma$ and an action $a \in A$ such that τM_a^σ is non-empty. We need to show that $\ell_V(\tau M_a^\sigma) = \ell_V(\tau)E_a$. Towards this, consider the equalities,

$$\ell_V(\tau M_a^\sigma) = \mathsf{end}(\tau M_a^\sigma) = \mathsf{end}(\tau M_a) = \mathsf{end}(\tau)E_a = \ell_V(\tau)E_a,$$

where the first and last equalities follow from the definition of ℓ_V, the second follows from the observation that the extended strategy tree $\mathscr{E}\mathscr{T}_G^\sigma$ is strategic (a consequence of Proposition **??**), and the third equality follows from the definition of M. Moreover, since τM_a is exactly the set of histories $\{\tau a v \mid v \in \ell_V(\tau)E_a\}$, it follows from the above equalities that $|\tau M_a^\sigma| = |\tau M_a| = |\ell_V(\tau)E_a|$, and therefore the condition **V.1** is satisfied by the witness.

- **V.2**: Consider the premise of **V.2**, that is, transitions $(\tau_1, a, \tau_2), (\tau_1', a', \tau_2') \in M^\sigma$ and an agent $i \in [n]$ such that $\tau_1 \approx_i^\sigma \tau_1'$. Also, since $\tau_1 \approx_i^\sigma \tau_1'$, its follows from the definition of $\approx_i^\sigma$ that $\mathsf{view}_i(\tau_1) = \mathsf{view}_i(\tau_2)$. Now consider the following:

  $\ell_V(\tau_2)^{\downarrow i} = \ell_V(\tau_2')^{\downarrow i}$ and $a^{\downarrow i} = a'^{\downarrow i}$

  $\Leftrightarrow \mathsf{end}(\tau_2)^{\downarrow i} = \mathsf{end}(\tau_2')^{\downarrow i}$ and $a^{\downarrow i} = a'^{\downarrow i}$ (By definition of ℓ_V)

  $\Leftrightarrow \mathsf{view}_i(\tau_1)\, a^{\downarrow i}\mathsf{end}(\tau_2)^{\downarrow i} = \mathsf{view}_i(\tau_1')\, a'^{\downarrow i}\mathsf{end}(\tau_2')^{\downarrow i}$ (By the assumption above)

  $\Leftrightarrow \mathsf{view}_i(\tau_1 a\ \mathsf{end}(\tau_2)) = \mathsf{view}_i(\tau_1' a'\ \mathsf{end}(\tau_2'))$

  $\Leftrightarrow \mathsf{view}_i(\tau_2) = \mathsf{view}_i(\tau_2')$ (By definition of $M_a^\sigma, M_{a'}^\sigma$)

  $\Leftrightarrow \tau_2 \approx_i^\sigma \tau_2'$. (By definition of $\approx^\sigma$)

  This shows that the condition **V.2** is satisfied by the witness.

- **Q.1**,**Q.2**: The conditions **Q.1**,**Q.2** are verbatim the defining conditions of the strong acceptance run f_Q, with every occurrence of ν replaced with ℓ_V. This replacement is sound, since $\nu(\tau) = \mathsf{end}(\tau) = \ell_V(\tau)$ for all histories $\tau \in H^\sigma$.

Now for the deterministic case, if σ is deterministic and if the MaLTS $\mathscr{W}$ is a witness, then note the following: firstly, the MaTS underlying the witness $\mathscr{W}$ is identical to the MaTS underlying extended strategy tree $\mathscr{E}\mathscr{T}_G^\sigma$; secondly, by Proposition **??**, whenever σ

is deterministic the MaTS underlying the extended strategy tree $\mathscr{E}\mathscr{T}^{\sigma}_{G}$ is also deterministic. It follows from these two observations that the the witness $\mathscr{W}$ is also deterministic. This completes the proof of this direction.

($\Leftarrow$) To see the reverse direction of the theorem, assume the existence of a witness $\mathscr{W} = (S,R,\{\sim_i\}_{i\in[n]},\{\ell_V,\ell_Q\},s_\varepsilon)$ for the game $(G,\mathscr{Q})$.

Let $\widetilde{S}$ denote the set of paths in $\mathscr{W}$ that begin at the start state s_ε, and let $f_V^- : \widetilde{S} \rightarrow (VA)^*V$ be the 'extension' of the labelling function ℓ_V to finite paths $s_0a_1s_1a_2...$ in $\widetilde{S}$, defined as follows: $f_V^-(s_0a_1s_1a_2...) = \ell_V(s_0)a_1\ell_V(s_1)a_2....$

Also, let H' denote the set of sequences $f_V^-(\widetilde{S})$, and let $f_V : \widetilde{S} \rightarrow H'$ denote the function obtained by restricting the co-domain of f_V^- to H'.

Consider an MaLTS $\mathscr{E}\mathscr{T}'_G = (H',M',\{\approx'_i\}_{i\in[n]},\nu',v_\varepsilon)$ such that, $H' = f_V(\widetilde{S})$, the MaTS $(H',M',\{\approx'_i\}_{i\in[n]},v_\varepsilon)$ is the substructure of the game tree $\mathscr{T}_G$ induced by H', and $\nu' : H' \rightarrow V$ is a labelling defined as $\nu'(\tau) = \mathsf{end}(\tau)$ for all $\tau \in H'$.

We claim that $\mathscr{E}\mathscr{T}'_G$ is the extended strategy tree of some winning joint strategy of the game $(G,\mathscr{Q})$, and additionally that, if $\mathscr{W}$ is deterministic, then $\mathscr{E}\mathscr{T}'_G$ is the extended strategy tree of some winning deterministic joint strategy.

We begin with some observations about the objects defined above.

Lemma 4.2.2.

1. *H' consists of histories of game graph G.*

2. *f_V is a bijection.*

Proof. For the first part, we need to show that for any path $\widetilde{s} = s_0a_1s_1a_2...s_k \in \widetilde{S}$, the sequence $\ell_V(s_0)a_1\ell_V(s_1)a_2..\ell_V(s_k)$ that belongs to H' is a history. We prove this claim by induction on the length k of the path. For the base case $k = 0$, firstly observe that $s_0 = s_\varepsilon$, since all paths in $\widetilde{S}$ begin at the state s_ε. Now observe that by the $\mathsf{Strategy}$-

Simulation condition of the witness $\mathscr{W}$, $\ell_V(s_\varepsilon) = v_\varepsilon$ holds, and therefore it follows that $\ell_V(s_0) = v_\varepsilon$, a history. For the inductive case k, assume that the claim is true for histories of length less than k. By this assumption, it follows that $\ell_V(s_0)a_1\ell_V(s_1)a_2..\ell_V(s_{k-1})$ is a history in H'. Now observe that $s_k \in s_{k-1}R_{a_k}$ holds due to definition of the path $\widetilde{s}$, and since $s_{k-1}R_{a_k} \neq \emptyset$, it follows from the Strategy-Simulation condition on the witness $\mathscr{W}$, that $\ell_V(s_{k-1}R_{a_k}) = \ell_V(s_{k-1})E_{a_k}$. Since $s_k \in s_{k-1}R_{a_k}$, it further follows that $\ell_V(s_k) \in \ell_V(s_{k-1}R_{a_k}) = \ell_V(s_{k-1})E_{a_k}$. It follows from this observation and the induction hypothesis, that the sequence $\ell_V(s_0)a_1\ell_V(s_1)a_2..\ell_V(s_k)$ is a path in the game graph.

For the second part, assume for a contradiction that f_V is not a bijection. Then there exist distinct paths $\widetilde{s}_1, \widetilde{s}_2$ such that $f_V(\widetilde{s}_1) = f_V(\widetilde{s}_2) = \tau$. It follows that there must exist distinct prefixes $\widetilde{s}as_1$ and $\widetilde{s}as_2$ of the respective paths $\widetilde{s}_1$ and $\widetilde{s}_2$, such that $\widetilde{s}$ is the least common prefix of $\widetilde{s}_1$ and $\widetilde{s}_2$. Let $s = \mathsf{end}(\widetilde{s})$. Now note that the states $s_1, s_2 \in sR_a$ are distinct, due to the prefixes $\widetilde{s}as_1$, $\widetilde{s}as_2$ being distinct. Also, since $f_V(\widetilde{s}_1) = f_V(\widetilde{s}_2)$ holds, it follows from the definition of f_V that $\ell_V(s_1) = \ell_V(s_2)$. Combining the above observations, we obtain that $|sR_a| < |\ell_V(sR_a)|$. Next, since $s_1, s_2 \in sR_a$, we have $sR_a \neq \emptyset$, which by the Strategy-Simulation condition implies $\ell_V(sR_a) = \ell_V(s)E_a$ and $|sR_a| = |\ell_V(s)E_a|$. Combining the above, we obtain $|sR_a| < |\ell_V(sR_a)| = |\ell_V(s)E_a| = |sR_a|$, a contradiction. □

Lemma 4.2.3. *For any histories $\tau, \tau' \in H'$, paths $\widetilde{s} \in f_V^{-1}(\tau), \widetilde{s}' \in f_V^{-1}(\tau')$ and agent $i \in [n]$, if $\mathsf{view}_i(\tau) = \mathsf{view}_i(\tau')$), then $\mathsf{end}(\widetilde{s}) \sim_i \mathsf{end}(\widetilde{s}')$.*

Proof. Consider paths and histories as above, and let $\widetilde{s} = s_0a_1..a_ms_m$ and $\widetilde{s}' = s'_0a'_1..a'_{m'}s'_{m'}$. Now assume that the premise $\mathsf{view}_i(\tau) = \mathsf{view}_i(\tau')$ holds.

We show the claim $\mathsf{end}(\widetilde{s}) \sim_i \mathsf{end}(\widetilde{s}')$, or equivalently, $s_m \sim_i s'_{m'}$, by induction on the sum $m + m'$. Firstly observe that for the base case, $\mathsf{end}(s_\varepsilon) \sim_i \mathsf{end}(s_\varepsilon)$ holds true. Next, since $\mathsf{view}_i(\tau) = \mathsf{view}_i(\tau')$, or equivalently, $\mathsf{view}_i(f_V(\widetilde{s})) = \mathsf{view}_i(f_V(\widetilde{s}'))$ holds, it follows from the definition of view_i and f_V that $\ell_V(s_m)^{\downarrow i} = \ell_V(s'_{m'})^{\downarrow i}$, $a_m^{\downarrow i} = a_{m'}'^{\downarrow i}$ and $\mathsf{view}_i(f_V(s_0a_1..a_{m-1}s_{m-1}) = \mathsf{view}_i(f_V(s'_0a'_1..a'_{m'-1}s'_{m'-1})$. By induction hypothesis, we obtain that $s_{m-1} \sim_i s'_{m'-1}$. Moreover, since $\widetilde{s}, \widetilde{s}'$ are paths in $\widetilde{S}$, there exists transitions $(s_{m-1}, a_m, s_m), (s'_{m'-1}, a'_{m'}, s'_{m'}) \in R$.

Therefore we have transitions $(s_{m-1}, a_m, s_m), (s'_{m'-1}, a'_{m'}, s'_{m'}) \in R$ such that $s_{m-1} \sim_i s'_{m'-1}$, $a_m^{\downarrow i} = a'^{\downarrow i}_{m'}$ and $\ell_V(s_m)^{\downarrow i} = \ell_V(s'_{m'})^{\downarrow i}$. It follows from the $\mathsf{Indistinguishability}$ condition of the witness $\mathscr{W}$ that $s_m \sim_i s'_{m'}$. □

Lemma 4.2.4. *For any history $\tau \in H'$, path $\widetilde{s} \in f_V^{-1}(\tau)$ and action $a \in A$, we have $\mathsf{end}(\tau M'_a) = \ell_V(\mathsf{end}(\widetilde{s}) R_a)$.*

Proof. Consider a history $\tau \in H'$, path $\widetilde{s} \in f_V^{-1}(\tau)$, and action a. Then,

$$\begin{aligned}
&\mathsf{end}(\tau M'_a) \\
&= \{\mathsf{end}(\tau a v) \mid \tau a v \in \tau M'_a\} && \text{(By definition of } \tau M'_a) \\
&= \{\mathsf{end}(\tau a v) \mid \exists s \in S : \widetilde{s} a s \in \widetilde{S} \text{ and } f_V(\widetilde{s} a s) = \tau a v\} && \text{(Since } \tau M'_a \subseteq H') \\
&= \{\mathsf{end}(\tau a v) \mid \exists s \in S : s \in \mathsf{end}(\widetilde{s}) R_a \text{ and } f_V(\widetilde{s} a s) = \tau a v\} && \text{(Since } \widetilde{s} a s \text{ is a path in } \mathscr{W}) \\
&= \{v \mid \exists s \in S : s \in \mathsf{end}(\widetilde{s}) R_a \text{ and } \ell_V(s) = v\} && \text{(By definition of } f_V, \mathsf{end}) \\
&= \{\ell_V(s) \mid s \in \mathsf{end}(\widetilde{s}) R_a\} \\
&= \ell_V(\mathsf{end}(\widetilde{s}) R_a)
\end{aligned}$$

□

Lemma 4.2.5. *If there exists a path $\tau_0 a_0 \tau_1 a_1 \ldots$ in $\mathscr{E}\mathscr{T}'_G$, then there exists a path $\mathsf{end}(\widetilde{s}_0)\ a_0\ \mathsf{end}(\widetilde{s}_1)\ a_1 \ldots$ in $\mathscr{W}$ with $\widetilde{s}_k \in f_V^{-1}(\tau_k)$ for each index k.*

Proof. To see this, consider an infinite path $\tau_0 a_0 \tau_1 a_1 \ldots$ in $\mathscr{E}\mathscr{T}'_G$, and let $\widetilde{s}_k$ be the unique path in that satisfies $\widetilde{s}_k \in f_V^{-1}(\tau_k)$. Since f_V is a bijection, such a unique path exists. It is an easy inductive consequence of the definition of f_V, that $\mathsf{end}(\widetilde{s}_0) a_0 \mathsf{end}(\widetilde{s}_1) a_1 \ldots$ is an infinite path in the witness $\mathscr{W}$. □

Returning to our claim, we first show that $\mathscr{E}\mathscr{T}'_G$ is the extended strategy tree of some joint strategy. Towards this, firstly observe that the labelling function ν' of $\mathscr{E}\mathscr{T}'_G$ is defined as required by an extended strategy tree. Therefore it suffices to show that the

MaTS $\mathscr{H} = (H', M', \{\approx'_i\}_{i\in[n]}, v_\varepsilon)$, is a MaTS underlying the extended strategy tree of some joint strategy. According to Proposition **??**, this is equivalent to showing that $\mathscr{H}$ is non-terminal, strategic and uniform. We argue these next.

- For $\mathscr{H}$ to be non-terminal, every history in H' must have a prolongation in H'. To see this, consider a history $\tau \in H'$ along with the path $\widetilde{s} \in f_V(\tau)$. By $\mathsf{Non\text{-}Termination}$ condition of the witness $\mathscr{W}$, it must be the case that state $\mathsf{end}(\widetilde{s}) \in S$ has some outgoing transition, say $(\mathsf{end}(\widetilde{s}), a, s)$. It is easy to see from the definition of f_V that, the history $f_V(\widetilde{s}as)$ is a prolongation of τ.

- For $\mathscr{H}$ to be strategic, it must satisfy two conditions: first, that $v_\varepsilon \in H'$; and second that, for any $\tau \in H'$ and action a, if $\tau M_a \cap H' \neq \emptyset$, then $\tau M_a \subseteq H'$.

  For the first part, observe that by first part of the $\mathsf{Strategy\text{-}Simulation}$ condition satisfied by $\mathscr{W}$, we have $\ell_V(s_\varepsilon) = v_\varepsilon$. Now since s_ε is a path in $\widetilde{S}$, it follows from the definition of f_V that $f_V(s_\varepsilon) = \ell_V(s_\varepsilon) = v_\varepsilon$. Moreover, since $H' = f_V(\widetilde{S})$, we have that $v_\varepsilon \in H'$.

  For the second part, consider a history $\tau \in H'$, the path $\widetilde{s} \in f_V(\tau)$ and an action a such that, $\tau M_a \cap H' \neq \emptyset$. Since $M'_a = M_a \cap (H' \times H')$, by its definition, the premise $\tau M_a \cap H' \neq \emptyset$ can be restated as $\tau M'_a \neq \emptyset$. Moreover, since $\ell_V(\mathsf{end}(\widetilde{s})R_a) = \mathsf{end}(\tau M'_a)$ holds by Lemma **??**, we obtain that $\mathsf{end}(\widetilde{s})R_a \neq \emptyset$. Now if $\mathsf{end}(\widetilde{s})R_a \neq \emptyset$, then the $\mathsf{Strategy\text{-}Simulation}$ condition satisfied by the witness $\mathscr{W}$ implies that, $\ell_V(\mathsf{end}(\widetilde{s})R_a) = \ell_V(\mathsf{end}(\widetilde{s}))E_a$. Now consider the following equalities:

$$\begin{aligned}
\tau M'_a &= \{\tau a v \mid v \in \mathsf{end}(\tau M'_a)\} && \text{(By definition of } \tau M'_a\text{)}\\
&= \{\tau a v \mid v \in \ell_V(\mathsf{end}(\widetilde{s})R_a)\} && \text{(By Lemma ??)}\\
&= \{\tau a v \mid v \in \ell_V(\mathsf{end}(\widetilde{s}))E_a\} && \text{(Since } \ell_V(\mathsf{end}(\widetilde{s})R_a) = \ell_V(\mathsf{end}(\widetilde{s}))E_a\text{)}\\
&= \{\tau a v \mid v \in \mathsf{end}(f_V(\widetilde{s}))E_a\} && \text{(By definition of } f_V \text{ and } \mathsf{end}\text{)}\\
&= \{\tau a v \mid v \in \mathsf{end}(\tau)E_a\} && \text{(By definition of } \widetilde{s}\text{)}\\
&= \tau M_a && \text{(By definition of } M\text{)}
\end{aligned}$$

Now since M'_a is the transition relation of the MaTS $\mathscr{H}$, it follows that $\tau M'_a \subseteq H'$, and therefore using the above equality, we obtain $\tau M_a \subseteq H'$.

- For $\mathscr{H}$ to be uniform, we need to show the following conditions:

  1. $\forall \tau_1, \tau_2 \in H', \forall i \in [n]$: if $\tau_1 \approx'_i \tau_2$, then $\{a^{\downarrow i} \in A_i \mid \tau_1 M'_a \neq \emptyset\} = \{a^{\downarrow i} \in A_i \mid \tau_2 M'_a \neq \emptyset\}$, and
  2. $\forall \tau \in H' : \{a \in A \mid \tau M'_a \neq \emptyset\} = \prod_{i \in [n]} \{a^{\downarrow i} \mid \tau M'_a \neq \emptyset\}$.

  Towards this, firstly observe that for any history $\tau \in H'$, path $\widetilde{s} \in f_V^{-1}(\tau)$ and action $a \in A$, we have by **??** that $\mathsf{end}(\widetilde{s})R_a \neq \emptyset$, if and only if, $\tau M'_a \neq \emptyset$. Therefore any use of the predicate $\mathsf{end}(\widetilde{s})R_a \neq \emptyset$, may be replaced by $\tau M'_a \neq \emptyset$.

  Now for the first condition, consider any histories $\tau_1, \tau_2 \in H'$ that satisfy $\tau_1 \approx'_i \tau_2$, and paths $\widetilde{s_1} \in f_V^{-1}(\tau_1)$ and $\widetilde{s_2} \in f_V^{-1}(\tau_2)$. Since $\approx'_i$ is the sub-relation of $\approx_i$ induced by histories H', it follows from the definition of $\approx_i$ that $\mathsf{view}_i(\tau_1) = \mathsf{view}_i(\tau_2)$. By Lemma **??**, it follows that that $\mathsf{end}(\widetilde{s_1}) \sim_a \mathsf{end}(\widetilde{s_2})$. Since $\mathsf{end}(\widetilde{s_1}) \sim_a \mathsf{end}(\widetilde{s_2})$ holds, it follows from the Uniformity-1 condition satisfied the witness $\mathscr{W}$ that, $\{a^{\downarrow i} \in A_i \mid \mathsf{end}(\widetilde{s_1})R_a \neq \emptyset\} = \{a^{\downarrow i} \in A_i \mid \mathsf{end}(\widetilde{s_2})R_a \neq \emptyset\}$. The desired conclusion is obtained by replacing the predicates $\mathsf{end}(\widetilde{s_1})R_a \neq \emptyset$ and $\mathsf{end}(\widetilde{s_2})R_a \neq \emptyset$, by $\tau_1 M'_a \neq \emptyset$ and $\tau_2 M'_a \neq \emptyset$ respectively.

  For the second condition, consider any history $\tau \in H'$, path $\widetilde{s} \in f_V^{-1}(\tau)$ and the state $\mathsf{end}(\widetilde{s}) \in S$. Observe that due to Uniformity-2 condition, it must be the case that $\{a \in A \mid \mathsf{end}(\widetilde{s})R_a \neq \emptyset\} = \prod_{i \in [n]} \{a^{\downarrow i} \mid \mathsf{end}(\widetilde{s})R_a \neq \emptyset\}$. The desired conclusion follows by replacing predicates as done before.

Therefore $\mathscr{E}\mathscr{T}'_G$ is the extended strategy tree of some joint strategy. Let σ be a joint strategy such that $\mathscr{E}\mathscr{T}^{\sigma}_G = \mathscr{E}\mathscr{T}'_G$. We argue next that if $\mathscr{W}$ is deterministic, then σ is also deterministic. To see this, assume for a contradiction that $\mathscr{W}$ is deterministic and that σ is not deterministic. Since σ is a joint strategy, we have that $\mathscr{E}\mathscr{T}'_G$ is non-terminal, strategic and uniform. It follows from Proposition **??** that $\mathscr{E}\mathscr{T}'_G$ is not deterministic. Therefore, there must exist a history $\tau \in H'$ and distinct actions a_1, a_2 that satisfy $\tau M'_{a_1} \neq \emptyset$ and $\tau M'_{a_2} \neq \emptyset$. Let $\widetilde{s}$ be a path such that $f_V(\widetilde{s}) = \tau$. Using Lemma **??**, it follows that there exist distinct actions a_1, a_2 at the state $\mathsf{end}(\widetilde{s})$, that satisfy $\mathsf{end}(\widetilde{s})T_{a_1} \neq \emptyset$ and $\mathsf{end}(\widetilde{s})T_{a_2} \neq \emptyset$, which is a contradiction to $\mathscr{W}$ being deterministic.

For our main claim to be true, it remains to show that the joint strategy σ is also winning, or equivalently, that the strategy tree of σ is strongly accepted by the μ-automaton $\mathscr{Q}$. Towards this, recall that the strategy tree of σ is by definition the LTS underlying the extended strategy tree $\mathscr{E}\mathscr{T}'_G$. Now let $\mathscr{T}'_G = (H', M', \nu', \nu_\varepsilon)$ be the LTS underlying the extended strategy tree $\mathscr{E}\mathscr{T}'_G$, and consider a function $f_Q : H' \to Q$ such that, for all histories $\tau \in H'$ and paths $\widetilde{s} \in f_V^{-1}(\tau)$, we have $f_Q(\tau) = \ell_Q(\mathsf{end}(\widetilde{s}))$. The function f_Q is well-defined, since f_V is a bijection.

We claim that LTS $\mathscr{T}'_G$ is strongly accepted by $\mathscr{Q}$ via the strong acceptance run f_Q.

Towards proving this, we state an observation about the function f_Q.

Lemma 4.2.6. *For any history $\tau \in H'$, path $\widetilde{s} \in f_V^{-1}(\tau)$ and action $a \in A$, we have $f_Q(\tau M'_a) = \ell_Q(\mathsf{end}(\widetilde{s})R_a)$.*

Proof. Consider a history $\tau \in H'$, path $\widetilde{s} \in f_V^{-1}(\tau)$ and action a. Then,

$$
\begin{aligned}
& f_Q(\tau M'_a) \\
&= \{f_Q(\tau a v) \mid \exists s \in S : \widetilde{s}as \in \widetilde{S} \text{ and } f_V(\widetilde{s}as) = \tau a v\} && \text{(By definition of } \tau M'_a, f_V) \\
&= \{\ell_Q(\mathsf{end}(\widetilde{s}as)) \mid \exists s \in S : \widetilde{s}as \in \widetilde{S}\} && \text{(By definition of } f_Q) \\
&= \{\ell_Q(\mathsf{end}(\widetilde{s}as)) \mid \exists s \in S : s \in \mathsf{end}(\widetilde{s})R_a\} && \text{(Since } \widetilde{s}as \text{ is a path in } \widetilde{S}) \\
&= \{\ell_Q(s) \mid \exists s \in S : s \in \mathsf{end}(\widetilde{s})R_a\} && \text{(By definition of end)} \\
&= \ell_Q(\mathsf{end}(\widetilde{s})R_a)
\end{aligned}
$$

□

We show next that f_Q satisfies the two conditions that are necessary for a strong acceptance run.

- The first condition requires that $\{(a, f_Q(\tau M'_a)) \mid a \in A\} \in \delta(f_Q(\tau), \nu'(\tau))$, for any history $\tau \in H'$.

  To see this, consider a history $\tau \in H'$ and a path $\widetilde{s} \in f_V^{-1}(\tau)$. Note that by the $\mathscr{Q}$-Simulation condition (or condition **Q.1**) from the definition of the witness $\mathscr{W}$, we have that $\{(a, \ell_Q(\mathsf{end}(\widetilde{s})R_a)) \mid a \in A\} \in \delta(\ell_Q(\mathsf{end}(\widetilde{s})), \ell_V(\mathsf{end}(\widetilde{s})))$. Now from Lemma **??** and the definition of f_Q it follows that $\mathsf{end}(\widetilde{s})R_a) = f_Q(\tau M'_a)$. Also, from definition of f_Q, f_V and ν', it follows that $f_Q(\tau) = \ell_Q(\mathsf{end}(\widetilde{s}))$ and $\nu'(\tau) = \ell_V(\mathsf{end}(\widetilde{s}))$. The desired conclusion $\{(a, f_Q(\tau M'_a)) \mid a \in A\} \in \delta(f_Q(\tau), \nu'(\tau))$ follows from these observations.

- The second condition requires that for any infinite path $\tau_0 a_0 \tau_1 a_1 ...$ in the LTS $\mathscr{T}'_G$, the priority sequence $\Omega(f_Q(\tau_0))\Omega(f_Q(\tau_1))...$ must satisfy the parity condition.

  To see this, consider an infinite path $\tau_0 a_0 \tau_1 a_1 ...$ in $\mathscr{H}'$. It follows from Lemma **??** that there exists a $\mathsf{end}(\widetilde{s}_0)a_0\mathsf{end}(\widetilde{s}_1)a_1 ...$ path in the witness $\mathscr{W}$ such that $\widetilde{s}_k \in f_V^{-1}(\tau_k)$ for each index k. By the $\mathscr{Q}$-Parity condition (or condition **Q.2**) satisfied by the witness $\mathscr{W}$, it follows that the priority sequence $\Omega(\ell_Q(\mathsf{end}(\widetilde{s}_0)))\Omega(\ell_Q(\mathsf{end}(\widetilde{s}_1)))...$ satisfies the parity condition. By the definition of f_Q, it follows that the above priority sequence is identical to the priority sequence $\Omega(f_Q(\tau_0))\Omega(f_Q(\tau_1))...$, and the desired result follows.

This completes the proof of the witness theorem. □

Having defined the notion of a witness we show next that it is easy to verify if a MaLTS is a witnesses.

Theorem 4.2.7 (Verifiability Theorem). *The problem of deciding if a finite first-order structure is a (deterministic) witness of the imperfect information game $(G, \mathscr{Q})$, is in* NLogSpace.

Proof. Let the term *basic conditions*, refer to the conditions **S.1**, **S.2**, **V.1**, **V.2** and **Q.1** for being a witness, excluding the $\mathscr{Q}$-Parity condition. Since the basic conditions are first-order formulae, the procedure for checking basic conditions, draws its complexity from the data complexity of model-checking structures for a fixed first-order formula. This is known to be in NLOGSPACE. Therefore if a structure $\mathscr{W}$ satisfies the basic conditions, then we may assume that the structure $\mathscr{W}$ is of the form $(S,R,(\sim_i)_{i\in[n]},\{\ell_V,\ell_Q\},s_\varepsilon)$ with the appropriate signature.

We show next that checking for the $\mathscr{Q}$-Parity condition, which is a second-order logic formula, is also NLOGSPACE. Towards this, we observe that the $\mathscr{Q}$-Parity condition admits the following alternative characterization: A structure $\mathscr{W}$ that satisfies the basic conditions also satisfies the $\mathscr{Q}$-Parity condition, if and only if, there are no 'odd simple cycles' in $\mathscr{W}$. A simple cycle $s_0a_1...a_ms_m$ is said to be *odd*, if the minimum priority in $\{\Omega(\ell_Q(s_i)) \mid 0 \leq i \leq m\}$ is odd. For a proof of this, observe that if there exists an odd simple cycle $s_0a_1...a_ms_m$ with $s_o = s_m$ in $\mathscr{W}$, then the priority sequence $(\Omega(\ell_Q(s_0))\Omega(\ell_Q(s_1))..\Omega(\ell_Q(s_m)))^\omega$ corresponding to the infinite path $\widetilde{s} = (s_0a_1...a_m)^\omega$ is easily seen to violate the parity condition. Conversely, assume that there are no odd simple cycles in $\mathscr{W}$, and consider an arbitrary infinite path $s_0a_1...$ in $\mathscr{W}$. Consider its suffix $s_ia_{i+1}s_{i+1}a_{i+2}..$, such that every state in the suffix appears infinitely often, and consider consecutive segments of this infinite path, that are simple cycles; since there are only finitely many states in $\mathscr{W}$, such a segmentation exists. It is easy to see that if none of the segments are odd, then the path satisfies the parity condition.

Checking the existence of an odd simple cycle can be done by a non-deterministic algorithm in space $\mathscr{O}(\log(|\mathscr{W}|))$. Therefore the algorithm to check whether a finite structure $\mathscr{W}$ is a witness for the imperfect information game $(G,\mathscr{Q})$ is in NLOGSPACE. $\square$

Next we define a transformation on MaLTS's, called a 'g-retract'.

Retracts of MaLTS's Consider an MaLTS $\mathscr{S} = (S,R,\{\sim_i\}_{i\in[n]},\{v_j\}_{j\in J},s_\varepsilon)$ and a function $g : S \to S$. Then,

- R^g denotes the 'composition' of R with g in the following sense: for any states $s_1, s_2 \in S$ and joint action $a \in A$, we have $(s_1, a, s_2) \in R^g$, if and only if, there exists a state $s \in S$ such that $(s_1, a, s) \in R$ and $s_2 = g(s)$. As before, for every action $a \in A$, R^g_a is the relation obtained by the composition $R_a \circ g$.

- A *g-path* is a sequence $s_0 a_0 s_1 a_1 ...$ such that $s_0 = g(s_\varepsilon)$ and $s_{m+1} \in s_m R^g_{a_m}$ for all $m \geq 0$. S^g denotes the set of states $s \in S$ such that, there exist a g-path that ends at s.

The *g-retract* of $\mathscr{S}$, denoted by $g(\mathscr{S})$, is the MaLTS $(S^g, R^g, \{\sim^g_i\}_{i \in [n]}, \{v^g_j\}_{j \in J}, s^g_\varepsilon)$, where $s^g_\varepsilon = g(s_\varepsilon)$, each functions v^g_j is the restriction of the function v_j to S^g and each relation $\sim^g_i$ is the restrictions of $\sim_i$ to S^g. The relation R^g here is the restriction of the relation R^g defined earlier to the domain S^g.

We note below an elementary consequence of the definition of g-retract that will be used in later chapters.

Proposition 4.2.8. *If g is a retraction of $\mathscr{S}$, then $S^g \subseteq g(S)$.*

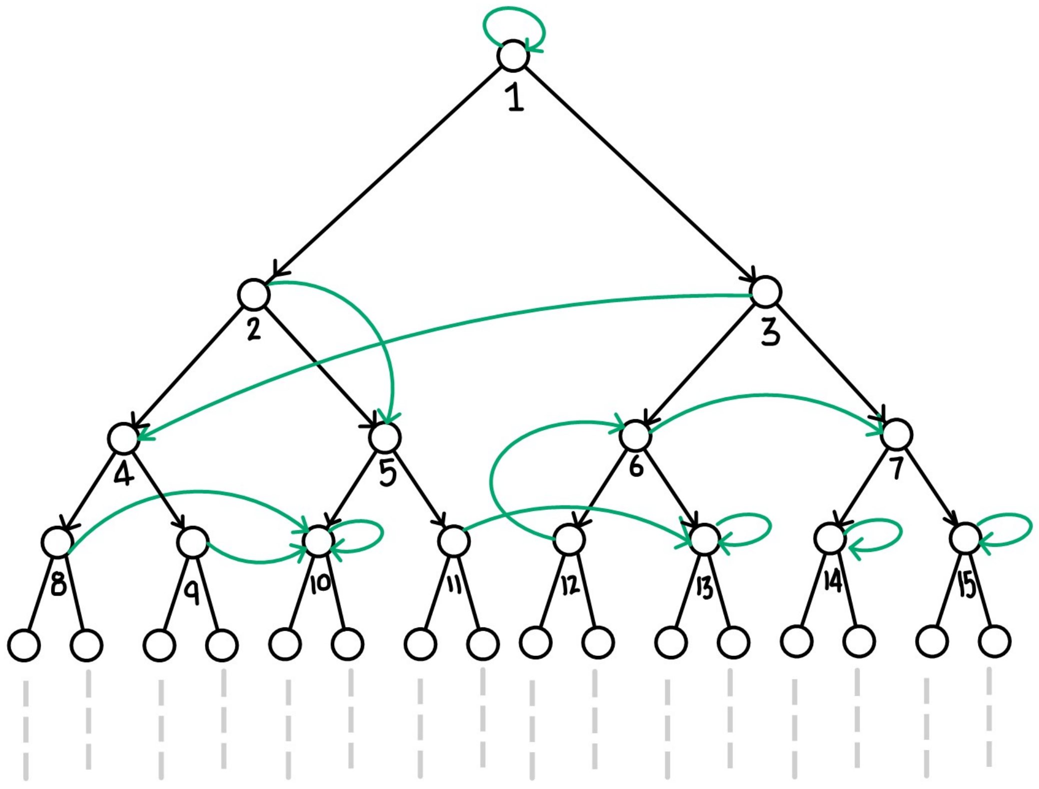
Proof. Note that for any $s \in S^g$, either $s = g(s_\varepsilon)$, or $s \in s' R^g_a$ for some $s' \in S, a \in A$ by virtue of being a state reachable by a g-path. In the second case, it follows from the definition of R^g_a that there exists a state $s'' \in S$ such that $g(s'') = s$. The claim follows from these observations. $\square$

A function g is said to be a *retraction* of the witness $\mathscr{W} = (S, R, \{\sim_i\}_{i \in [n]}, \{\ell_V, \ell_Q\}, s_\varepsilon)$, if it satisfies the following:

R.1 for any state $s \in S^g$ we have $\ell_V(s) = \ell_V(g(s))$ and $\ell_Q(s) = \ell_Q(g(s))$,

R.2 for any agent $i \in [n]$ and transitions $(s_1, a, s_2), (s'_1, a', s'_2) \in R^g$ with $s_1, s'_1 \in S^g$, if $s_1 \sim_i s'_1$, then, $s_2 \sim_i s'_2$, if and only if, $\ell_V(s_2)^{\downarrow i} = \ell_V(s'_2)^{\downarrow i}$ and $a^{\downarrow i} = a'^{\downarrow i}$, and

R.3 for any g-path $s_0 a_0 s_1 a_1 ...$, the priority sequence $\Omega(\ell_Q(s_0)), \Omega(\ell_Q(s_1)), ..$ satisfies the *parity condition*.



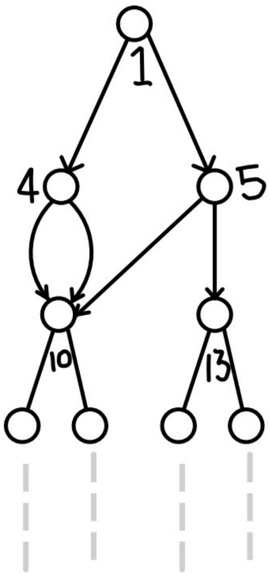


Figure 4.1: A MaLTS and its g-retract.

An MaLTS $\mathscr{W}$ is shown above along with a map g on the MaLTS represented by the green arrows. The g-retract $g(\mathscr{W})$ is given below.

We refer to **R.1,R.2** as *basic constraint* and **R.3** as the *parity constraint*.

Intuitively, a retraction g relates those states such that any incoming transition to s may be redirected to $g(s)$, while ensuring that the resulting structure remains a witness. In some sense a retraction relates states of the witness, where the joint actions chosen at a state may be reused at the other, while maintaining the winning status of the resulting joint strategy.

Theorem 4.2.9 (**Retraction Theorem**). *If g is a retraction of a (deterministic) witness $\mathscr{W}$ for the imperfect information game $(G,\mathscr{Q})$, then $g(\mathscr{W})$ is a (deterministic) witness for the imperfect information game $(G,\mathscr{Q})$.*

Proof. Let $\mathscr{W}$ be the imperfect information game $(G,\mathscr{Q})$, and let g be a retraction of $\mathscr{W}$. We claim that the MaLTS $g(\mathscr{W})$ is also a witness for $(G,\mathscr{Q})$.

First we state, without proof, some easy observations about $g(\mathscr{W})$.
(i). For any state $s \in S^g$ and joint action $a \in A$, $sR^g_a \neq \emptyset$, if and only if, $sR_a \neq \emptyset$.
(ii). For any states $s,s' \in S^g$ and agent $i \in [n]$, $s \sim^g_i s'$, if and only if, $s \sim_i s'$.
(iii). For any state $s \in S^g$ and joint action $a \in A$, we have $\ell_V(sR^g_a) = \ell_V(sR_a)$ and $\ell_Q(sR^g_a) = \ell_Q(sR_a)$.

Returning to our claim, first observe that the analogue of conditions **S.1**,**S.2** for the MaLTS $g(\mathscr{W})$, can be obtained by considering condition **S.1**,**S.2** for the witness $\mathscr{W}$, and replacing any predicates of the form $sR_a \neq \emptyset$ and $s \sim_i s'$ in this condition, by the predicate $sR^g_a \neq \emptyset$ and $s \sim^g_i s'$ respectively. By the observation (i),(ii) made above, such a replacement is sound.

To see that the condition **V.1** for being a witness holds for $g(\mathscr{W})$, we need to show that $\ell^g_V(s^g_\varepsilon)) = v_\varepsilon$, and that for any state $s \in S^g$ and joint action $a \in A$, $sR^g_a \neq \emptyset$ implies $\ell^g_V(sR^g_a) = \ell^g_V(s)E_a$ and $|sR^g_a| = |\ell^g_V(s)E_a|$. For the first part, note that $\ell^g_V(s^g_\varepsilon)) = \ell_V(g(s_\varepsilon)) = \ell_V(s_\varepsilon) = v_\varepsilon$, where the second equality follows from condition **R.1** and the other equalities follow from the definition of ℓ_V, ℓ^g_V and s^g_ε. For the second part, note that for any state $s \in S^g$ and joint action $a \in A$, if $sR^g_a \neq \emptyset$, then we have $\ell^g_V(sR^g_a) = \ell_V(sR^g_a) = \ell_V(sR_a) = \ell_V(s)E_a = \ell^g_V(s)E_a$, which follows from condition **V.1** for the witness $\mathscr{W}$ and the definition of the ℓ_V^g, R_a^g and observation (iii). The fact that $|sR_a^g| = |\ell_V^g(s)E_a|$ also follows easily, since each outgoing transition $(s,a,s') \in R$ at a state $s \in S^g$, is replaced by exactly one outgoing transition $(s,a,s'') \in R^g$. The condition **V.2** for $g(\mathscr{W})$, is verbatim the condition **R.2** from the definition of a retraction.

To see that the condition **Q.1** for being a witness holds for $g(\mathscr{W})$, we need to show that $\ell_Q^g(s_\varepsilon^g)) = q_\varepsilon$, and that for any state $s \in S^g$, $\{(a, \ell_Q(sR_a^g)) \mid a \in A\} \in \delta(\ell_Q^g(s), \ell_V^g(s))$. For the first part, note that $\ell_Q^g(s_\varepsilon^g)) = \ell_Q(g(s_\varepsilon)) = \ell_Q(s_\varepsilon) = q_\varepsilon$, where the second equality follows from condition **R.1** and the other equalities follow from the definition of ℓ_Q, ℓ_Q^g and s_ε^g. For the second part, consider the second part of condition **Q.1** for the witness $\mathscr{W}$, which states that for any state $s \in S$, $\{(a, \ell_Q(sR_a)) \mid a \in A\} \in \delta(\ell_Q(s), \ell_V(s))$. Since $S^g \subseteq S$ and since $\ell_Q(sR_a^g) = \ell_Q(sR_a)$ holds by observation (iii), it follows from the definitions of ℓ_Q^g, ℓ_V^g that for any state $s \in S^g$, $\{(a, \ell_Q(sR_a^g)) \mid a \in A\} \in \delta(\ell_Q^g(s), \ell_V^g(s))$. The condition **Q.2** for $g(\mathscr{W})$, is verbatim the condition **R.3** from the definition of a retraction. $\square$

Note that retractions are not 'composable', that is, if g_1, g_2 are retractions of a witness $\mathscr{W}$, then it is not always the case that $g_1 \circ g_2$ is a retraction of $\mathscr{W}$. However, if g_1 is a retraction of $\mathscr{W}$, and g_2 is a retraction of $g_1(\mathscr{W})$, the $g_1 \circ g_2$ is a retraction. This is not difficult to see, and we state this observation below for later use.

Proposition 4.2.10. *If g_1 is a retraction of the witness $\mathscr{W}$ and g_2 is a retraction of the witness $g_1(\mathscr{W})$, then $g_1 \circ g_2$ is a retraction of $\mathscr{W}$.*

4.3 Witnesses and Retractions for Local Specs.

In this section, we describe the notion of witness and retraction for imperfect information games that have local winning conditions given by priority labellings. Towards this, let us fix an imperfect information game $(G, \{\gamma_i\}_{i \in [n]})$, where $G = (V, E, v_\varepsilon)$ is a game graph and each $\gamma_i : V_i \to P$ is a priority labelling that maps the set of local states V_i of agent i to the priorities in $P = \{1, 2, .., |P|\}$.

Witnesses A $(\{V\}, \{A_i\}_{i\in[n]})$-MaLTS $\mathscr{W} = (S, R, \{\sim_i\}_{i\in[n]}, \{\ell_V\}, s_\varepsilon)$ is called a *witness* for the imperfect information game $(G, \{\gamma_i\}_{i\in[n]})$, if it satisfies the conditions **S.1**, **S.2**, **V.1**, **V.2** as defined earlier, and the condition given below:

S′.3. $\forall s_1, s_2 \in S, \forall i \in [n] : s_1 \sim_i s_2$ implies $\ell_V(s_1)^{\downarrow i} = \ell_V(s_2)^{\downarrow i}$.

S′.4. $\forall s_1, s_2 \in S, \exists i \in [n] : s_1 \not\sim_i s_2$. (Unique-Type)

Q′.1. for every path $s_0 a_1 s_1 a_2 ..$ and agent $i \in [n]$, the sequence $\gamma_i(\ell_V(s_0)^{\downarrow i}), \gamma_i(\ell_V(s_1)^{\downarrow i}), ..$ satisfies the parity condition. (γ-Parity)

A function g is said to be a *retraction* of the witness $\mathscr{W}$, if it satisfies the following:

R′.1 For any state $s \in S^g$ we have $\ell_V(s) = \ell_V(g(s))$,

R.2 for any agent $i \in [n]$ and transitions $(s_1, a, s_2), (s_1', a', s_2') \in R^g$ with $s_1, s_1' \in S^g$, if $s_1 \sim_i s_1'$, then, $s_2 \sim_i s_2'$, if and only if, $\ell_V(s_2)^{\downarrow i} = \ell_V(s_2')^{\downarrow i}$ and $a^{\downarrow i} = a'^{\downarrow i}$, and

R′.3 For any g-path $s_0 a_1 s_1 a_2 s_2 ...$ and agent $i \in [n]$, the priority sequence $\gamma_i(\ell_V(s_0)^{\downarrow i}), \gamma_i(\ell_V(s_1)^{\downarrow i}), \gamma_i(\ell_V(s_2)^{\downarrow i}), ..$ satisfies the parity condition.

Theorem 4.3.1 (Witness, Verifiability and Retraction Theorem-Local).

1. *For any imperfect information game $(G, \{\gamma_i\}_{i\in[n]})$, there exists a winning (deterministic) joint strategy for $(G, \{\gamma_i\}_{i\in[n]})$, if and only if, there exists a (deterministic) witness for $(G, \{\gamma_i\}_{i\in[n]})$.*

2. *There is an* NLOGSPACE *algorithm that decides if a given finite structure is a witness for the imperfect information game $(G, \{\gamma_i\}_{i\in[n]})$.*

3. *If g is a retraction of the (deterministic) witness $\mathscr{W}$ for the imperfect information*

game $(G, \{\gamma_i\}_{i\in[n]})$, then $g(\mathscr{W})$ is a (deterministic) witness for the imperfect information game $(G, \{\gamma_i\}_{i\in[n]})$.

Proof. The first proof follows along similar lines of the earlier witness theorem. For the forward direction, the canonical witness associated with the winning joint strategy σ is given by $\mathscr{W}' = (H^\sigma, M^\sigma, \{\approx_i^\sigma\}_{i\in[n]}, \{\ell_V\}, v_\varepsilon)$, which is identical to the witness $\mathscr{W}$ constructed in the earlier witness theorem but without the label ℓ_Q. The argument for the conditions $\mathbf{S.1}, \mathbf{S.2}, \mathbf{V.1}, \mathbf{V.2}$ to hold for $\mathscr{W}'$ follows from before. And argument for $\mathscr{W}'$ being deterministic for a deterministic joint strategy σ also follows from before. The fact that condition $\mathbf{S'.3}, \mathbf{S'.4}$ holds true is an easy consequence of the definition of $\approx_i^\sigma$ and the fact that $\ell_V = \mathsf{end}$. It remains to show that the condition $\mathbf{Q'.1}$ holds. Towards this, note that any path $\tau_0 a_1 \tau_2 a_2..$ in the witness $\mathscr{W}'$ that begins at the start state, by construction of the witness $\mathscr{W}'$, corresponds to the play $\mathsf{end}(\tau_0)a_1\mathsf{end}(\tau_0)a_2..$ that follows σ. Since σ is a winning joint strategy, we have that $\gamma_i(\mathsf{end}(\tau_0)^{\downarrow i})\gamma_i(\mathsf{end}(\tau_1)^{\downarrow i})..$ satisfies the parity condition for every agent $i \in [n]$, and since $\ell_V = \mathsf{end}$, we have that $\gamma_i(\ell_V(\tau_0)^{\downarrow i})\gamma_i(\ell_V(\tau_1)^{\downarrow i})..$ satisfies the parity condition for every agent $i \in [n]$. This shows that condition $\mathbf{Q'.1}$ holds for $\mathscr{W}'$.

For the reverse direction, consider a $\mathscr{W} = (S, R, \{\sim_i\}_{i\in[n]}, \{\ell_V\}, s_\varepsilon)$ for the game $(G, \{\gamma_i\}_{i\in[n]})$ and construct the extended strategy tree $\mathscr{E}\mathscr{T}'_G = (H', M', \{\approx'_i\}_{i\in[n]}, \mathsf{v}', v_\varepsilon)$ as in the earlier witness theorem. Note that this construction does not use any properties of the labelling ℓ_Q used there, and therefore can be admitted here. Now let $\mathscr{E}\mathscr{T}'_G$ be the extended strategy tree of some joint strategy σ. It remains to argue that σ is a winning joint strategy. Towards this consider a play $v_0 a_1 v_1 a_2..$ that follows σ. We need to show that $\gamma_i(v_0^{\downarrow i})\gamma_i(v_1^{\downarrow i})..$ satisfies the parity condition. Now since $\mathscr{E}\mathscr{T}'_G$ is the extended strategy tree of σ, it follows that the path $\tau_0 a_1 \tau_1 a_2..$, with each $\tau_k = v_0 a_1 v_1 a_2..v_k$ is a path in the $\mathscr{E}\mathscr{T}'_G$. By definition of f_V and the fact that f_V is a bijection, it follows that there exists a path $s_0 a_1 s_1 a_2..$ in $\mathscr{W}$ such that $\ell_V(s_0)a_1\ell_V(s_1)a_2.. = v_0 a_1 v_1 a_2..$, and therefore for any agent $i \in [n]$, the priority sequence $\gamma_i(\ell_V(s_0)^{\downarrow i})\gamma_i(\ell_V(s_1)^{\downarrow i}).. = \gamma_i(v_0^{\downarrow i})\gamma_i(v_1^{\downarrow i})...$ Since the witness $\mathscr{W}$ satisfies the condition

$\mathbf{Q'}$.1, the priority sequence $\gamma_i(\ell_V(s_0)^{\downarrow i})\gamma_i(\ell_V(s_1)^{\downarrow i})$.. satisfies the parity condition and the desired result follows.

The proofs of the second and third items above follow from arguments similar to the ones in the verifiability and retraction theorems proved in the previous section. □

4.4 Using the Retraction Approach

Retraction Approach for the Winning Strategy Problem We are now in a position to describe how we intend to use the retraction approach to solve winning strategy problems. The goal is to find classes $\mathscr{G}$ of imperfect information games, for which there exists a computable *size function* $f_{\mathscr{G}} : \mathbb{N}^2 \to \mathbb{N}$ such that the following hold:

For any game $(G, \mathscr{Q}) \in \mathscr{G}$ and canonical witnesses $\mathscr{W}$ of $(G, \mathscr{Q})$, the witness $\mathscr{W}$ admits a retraction g such that g-retract $g(\mathscr{W})$ is of size at most $f_{\mathscr{G}}(|G|, |\mathscr{Q}|)$.

Now if such a property is present for the class $\mathscr{G}$ of games, then the algorithm to solve winning strategy problem, operates by guessing a witness of size at most $f_{\mathscr{G}}(|G|, |\mathscr{Q}|)$ on an input $(G, \mathscr{Q}) \in \mathscr{G}$, and verifies it. If the verification holds, then return the LTS underlying the witness. It is not difficult to see that this LTS is the LTS-represenation of some winning strategy of the game. Since verifying a witness is in NLogSpace, the time complexity of this algorithm is NTime$(f_{\mathscr{G}}(|G|, |\mathscr{Q}|))$.

The correctness of the algorithm follows from the witness, verifiability and retraction theorems, and the fact that a winning strategy exists for a game, if and only if, there exists a canonical witness for the game. The case of games with local winning condition and that of deterministic winning strategy problem can be argued similarly. We consolidate these observations in the form of the following theorem.

Theorem 4.4.1 (Solvability Theorem for Games). *For a class $\mathscr{G}$ of imperfect information games, if there exists a computable function $f_{\mathscr{G}}$ such that every (determin-*

istic) canonical witness of every game $(G,W) \in \mathscr{G}$ *admits a retraction* g *such that* $|g(\mathscr{W})| \leq f_{\mathscr{G}}(|G|,|W|)$ *, then the (deterministic) winning strategy problem is solvable for the class* $\mathscr{G}$ *in* $\text{NT}\textsc{ime}(f_{\mathscr{G}}(|G|,|W|))$.

Retraction Approach for the Distributed Synthesis Problem As seen in Chapter 2, the (deterministic) distributed synthesis problem reduces in polynomial time to the (deterministic) winning strategy problem. Therefore to show that the solvability of the (determinitic) distributed synthesis problem for an architecture class $\mathscr{A}$ and a specification class $\mathscr{S}$, it suffices to show the *retraction criterion* stated below:

There exists a computable function f such that, for any architecture $\mathsf{Ar} \in \mathscr{A}$ and specification $W \in \mathscr{S}$. every (deterministic) canonical witness $\mathscr{W}$ of the game (G_{Ar},W), admits a retraction g such that $|g(\mathscr{W})| \leq f(|G_{\mathsf{Ar}}|,|W|)$.

It follows from the earlier Theorem **??** that the satisfaction of this criterion automatically gives an algorithm for solvability of the distributed synthesis problem, that is in $\text{NT}\textsc{ime}(f(|G_{\mathsf{Ar}}|,|W|))$.

Chapter 5

Synthesis for Global Specifications

In this chapter, we show the following:

1. The *deterministic distributed synthesis problem* is solvable for architectures that admit *weak-informedness ordering* and *global μ-automaton specifications*.
2. The *distributed synthesis problem* is solvable for architectures that admit *informedness ordering* and *global μ-automaton specifications*.

We divide our analyses of the (deterministic) distributed synthesis problem into the following steps:

1. First we show that the game graphs corresponding to architectures that admit (weak-) informedness ordering, satisfy the property that they contain no '(uniform-deterministic) fork-triples'.
2. Second we show that for any game $(G, \mathscr{Q})$ where G contains no (uniform-deterministic) fork-triples and $\mathscr{Q}$ is a global μ-automaton winning condition, every canonical (deterministic) witness of $(G, \mathscr{Q})$ is a 'modular witness'.
3. Third we show that there exists a computable function f such that, every modular

witness $\mathscr{W}$ of a game $(G,\mathscr{Q})$ of the kind mentioned above, admits a retraction g with $g(\mathscr{W})$ of size at most $f(|G|,|\mathscr{Q}|)$.

By the retraction criterion mentioned previously, the solvability of the above distributed synthesis problems follows.

As mentioned in Chapter 2, these architecture classes were solved previously using automata-theoretic approach[1]. Our aim is to use the retraction approach to solve these in a uniform fashion, and thereby show the robustness of this approach.

5.1 Architectures, Informedness and Fork-Triples

We begin by fixing some notation. Let $(G,\mathscr{Q})$ be an imperfect information game, where G is the game graph (V,E,v_ε) and $\mathscr{Q}$ is the μ-automaton $(Q,A,P,\delta,\Omega,q_\varepsilon)$. Let $\mathscr{T}_G$ denote the game tree specified by G.

We say that a game graph G contains a *fork-triple*, if there exists an induced substructure $\mathscr{T}'=(H',M',\{\approx'_i\}_{i\in[n]},v_\varepsilon)$ of the game tree $\mathscr{T}_G$ that satisfies the following:

1. $\mathscr{T}'$ contains no infinite paths and contains exactly three finite maximal paths of equal length.

2. There exist agents $i,j\in[n]$ and histories $\tau_1,\tau_2,\tau_3\in H'$ such that $\tau_1\approx'_j\tau_2\approx'_i\tau_3$ and $\tau_1\not\approx'_i\tau_2\not\approx'_j\tau_3$ hold.

We qualify a fork-triple $\mathscr{T}'$ as *uniform-deterministic*, if $\mathscr{T}'$ is uniform and deterministic.

We begin by showing that the criterion of existence of a fork-triple in a game graph is an effective one.

[1]The case of 'hierarchy ordering' mentioned in Chapter 2 is replaced here with 'informedness ordering', because the solution for hierarchy ordering uses the notion of views that excludes actions from the view.

Proposition 5.1.1. *There exists an* NLOGSPACE *algorithm that decides whether a game graph contains a (uniform-deterministic) fork-triple.*

Proof. We begin with the case of fork-triples. Firstly observe that if there exists fork-triple in the game graph G, then there exists a fork-triple in which the length of the maximal histories in the fork-triple is bounded by $|G|^3 \times n.2^3$. To see this, consider a fork-triple $\mathscr{T}' = (H', M', \{\approx'_i\}_{i\in[n]}, v_\varepsilon)$ with its three maximal histories denoted by τ_1, τ_2, τ_3. For each $k \in \{1,2,3\}$, let τ_k^ℓ denote the ℓ-length prefix of τ_k.

Now consider any triples of histories $(\tau_1^\ell, \tau_2^\ell, \tau_3^\ell)$, $(\tau_1^{\ell'}, \tau_2^{\ell'}, \tau_3^{\ell'})$ with $\ell < \ell'$, that satisfy the following: there exists an isomorphism f between the substructures of the fork-triple induced by the history-sets $\{\tau_1^\ell, \tau_2^\ell, \tau_3^\ell\}$ and $\{\tau_1^{\ell'}, \tau_2^{\ell'}, \tau_3^{\ell'}\}$, such that $f(\tau_k^\ell) = \tau_k^{\ell'}$ for all $k \in \{1,2,3\}$. Now if such triples exist, then one can obtain a shorter fork-triple by considering each path in the fork-triple that begins at τ_k^ℓ and ends at $\tau_k^{\ell'}$, and collapsing all histories that lie in between, to the state τ_k^ℓ. It is not difficult to argue that the resulting structure is also a fork-triple. Also there are at most $|G|^3 \times n.2^3$ many history-triples that induce non-isomorphic substructures; $|G|$-many labels for each history in the history-triple and for each agent in $[n]$, at most 2^3-many possibilities equivalence relations.

Having shown this bound on the fork-triple, we describe the algorithm that checks for existence of fork-triples. The algorithm proceeds by maintaining three pointers x_1, x_2, x_3 for storing states of the game graph and pointers to store two equivalence relations, say $\approx'_i$ and $\approx'_j$, on these three pointers. It begins by initializing each of the three pointers to the start vertex of the game graph, and performs at most $|G|^3 \times n.2^3$ iterations, where in each iteration, each of the pointers to the vertices are non-deterministically updated to a successor vertex in the game graph, and the equivalence relations due to $\approx'_i, \approx'_j$ on these new vertices are updated accordingly. If at one of these iterations, the pointers refer to histories $\tau_1, \tau_2, \tau_3 \in H'$ such that $\tau_1 \approx'_j \tau_2 \approx'_i \tau_3$ and $\tau_1 \not\approx'_i \tau_2 \not\approx'_j \tau_3$, then the algorithm outputs YES, else at the end of the iterations, it outputs NO. The correctness of this algorithm is easy to see, and since the algorithm only maintains three pointers to the states and two equivalence relations on these states, it requires only logarithmic space. The additional checks to ensure that the fork-triple is uniform-deterministic simply require us to do a constant-time check at each iteration, which again requires only logarithmic space. It follows that the algorithm in NLOGSPACE. $\square$

Now consider an architecture $\mathsf{Ar} = ([n]^+, \alpha)$ on a set of variables Z. Recall that for any agent $i \in [n]$, In_i denotes the set of input variables of agent i, Out_i denote the set of output variables of agent i.

Also recall that if G is a game graph that corresponds to the architecture Ar, then

- $V = \prod_{i \in [n]} V_i$, such that $V_i = \prod_{z \in \mathsf{In}_i} \{[z=0],[z=1]\}$ for each agent i,
- $v_\varepsilon \in V$ such that $v_\varepsilon^{\downarrow i \downarrow z} = [z=0]$ for each agent i and $z \in \mathsf{In}_i$,
- $A = \prod_{i \in [n]} A_i$, such that $A_i = \prod_{z \in \mathsf{Out}_i} \{[z=0],[z=1]\}$ for each agent i,
- $E \subseteq V \times A \times V$ is such that $(v,a,v') \in E$, if and only if, for any agents $i, j \in [n]$ and variable $z \in \mathsf{Out}_i \cap \mathsf{In}_j$, we have $a^{\downarrow i \downarrow z} = v'^{\downarrow j \downarrow z}$.

We say that an architecture Ar admits

- an *informedness ordering*[2], if there exists an enumeration $i_1, i_2, ... i_n$ of agents in $[n]$, such that $\mathsf{In}_{i_j} \cup \mathsf{Out}_{i_j} \supseteq \mathsf{In}_{i_{j+1}} \cup \mathsf{Out}_{i_{j+1}}$ holds for all indices $1 \leq j < n$.
- a *weak-informedness ordering*, if there exists an enumeration $i_1, i_2, ... i_n$ of agents in $[n]$, such that $\mathsf{In}_{i_j} \cup \mathsf{Out}_{i_j} \supseteq \mathsf{In}_{i_{j+1}}$ holds for all indices $1 \leq j < n$.

In either of the above cases, we call the enumeration $i_1, i_2, ... i_n$ as a (weak-)informedness ordering, and call the last element in this enumeration as the *least-informed agent*.

Next we show that the games corresponding to the above mentioned architectures satisfy the property that they contain no 'fork-triples'.

[2]Instead, the *hierarchy ordering* requires that $\mathsf{In}_{i_j} \supseteq \mathsf{In}_{i_{j+1}}$ holds for all indices $1 \leq j < n$

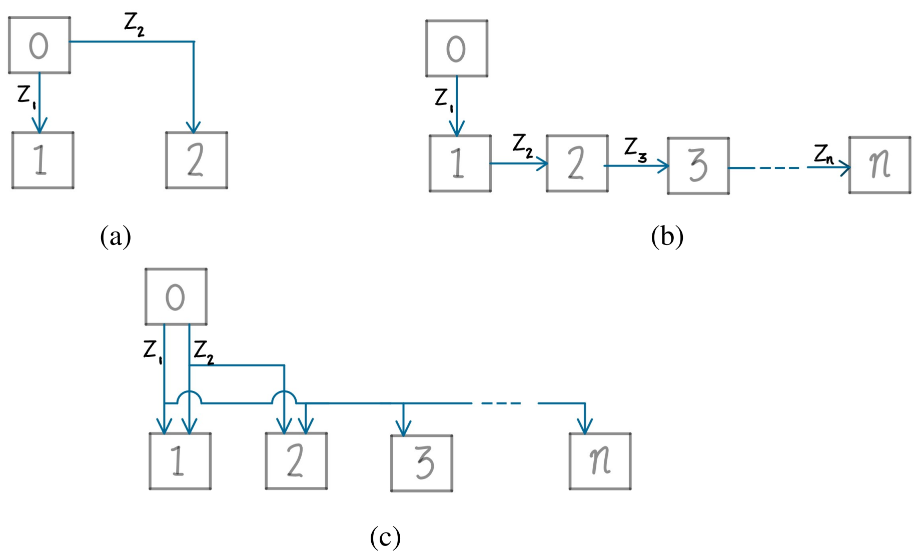


Figure 5.1: (Weak-)Informedness Architectures: Examples and Non-Examples.

The architecture (a) *admits neither a weak-informedness nor an informedness ordering. The architecture* (b) *admits a weak-informedness ordering, but not informedness ordering. The architecture* (c) *admits both weak-informedness and informedness ordering.*

Proposition 5.1.2. *Let Ar be an architecture that admits a (weak-)informedness ordering, and let G be a game graph that corresponds to the architecture Ar. Then, G contains no (uniform-deterministic) fork-triples.*

Proof. We argue both cases simultaneously. Assume for a contradiction that $\mathscr{T}' = (H', M', \{\approx'_i\}_{i\in[n]}, v_\varepsilon)$ is a a (uniform-deterministic) fork-triple of the game G with its three maximal histories given by τ_1, τ_2, τ_3. Let $\tau_1^\ell, \tau_2^\ell, \tau_3^\ell$ denote the ℓ-length prefix of the histories τ_1, τ_2, τ_3 respectively.

We argue inductively that for any $k, k' \in \{1,2,3\}$, length ℓ and agent i, if $\tau_k^\ell \approx'_i \tau_{k'}^\ell$, then $\tau_k^\ell \approx'_{i+1} \tau_{k'}^\ell$. It is easy to see that if this is true, then we obtain a contradiction to our assumption that $\mathscr{T}'$ is a fork-triple.

Now towards this, firstly consider the base case $\ell = 0$, and observe that in this case, we have $\tau_k^0 = \tau_{k'}^0 = v_\varepsilon$, and therefore the claim follows. For the inductive case, consider a length ℓ and assume that the claim holds true for histories of this length, and assume for a contradiction that the claim is false for the length $\ell+1$.

Let $\tau_k^{\ell+1} = \tau_k^\ell av$ and $\tau_{k'}^{\ell+1} = \tau_{k'}^\ell a'v'$. It follows from the claim being false for the length $\ell+1$, that $\tau_k^\ell av \approx_i' \tau_{k'}^\ell a'v'$ and $\tau_k^\ell av \not\approx_{i+1}' \tau_{k'}^\ell a'v'$. Since $\tau_k^\ell av \approx_i' \tau_{k'}^\ell a'v'$ holds, it follows from the synchronous perfect-recall property on the game tree that $\tau_k^\ell \approx_i' \tau_{k'}^\ell$, and it follows from the induction hypothesis that $\tau_k^\ell \approx_{i+1}' \tau_{k'}^\ell$.

Now since $\tau_k^\ell av \not\approx_{i+1}' \tau_{k'}^\ell a'v'$, it follows from the definition of $\approx_{i+1}'$ that either $a^{\downarrow i+1} \neq a'^{\downarrow i+1}$ or $v^{\downarrow i+1} \neq v'^{\downarrow i+1}$. The argument now splits into two cases, one for the weak-informedness case and the other for the informedness case:

- Weak-Informedness: In this case, since the fork-triple $\mathscr{T}'$ is uniform and deterministic, and since $\tau_k^\ell \approx_{i+1}' \tau_{k'}^\ell$ holds, it follows that the action of agent $i+1$ in the joint actions a and a' are identical, and so $a^{\downarrow i+1} \neq a'^{\downarrow i+1}$ is always false. Therefore it must be the case that $v^{\downarrow i+1} \neq v'^{\downarrow i+1}$ holds, or equivalently, that there exists a variable $z \in \mathsf{In}_{i+1}$ such that $v^{\downarrow i+1 \downarrow z} \neq v'^{\downarrow i+1 \downarrow z}$.

  Since $\mathsf{In}_i \cup \mathsf{Out}_i \supseteq \mathsf{In}_{i+1}$ holds by definition of weak-informedness, the variable z read by agent $i+1$, is read or written on by agent i. Therefore it follows from the construction of the game graph G_{Ar}, that either $a^{\downarrow i \downarrow z} \neq a'^{\downarrow i \downarrow z}$ or $v^{\downarrow i \downarrow z} \neq v'^{\downarrow i \downarrow z}$.

- Informedness: Since either $a^{\downarrow i+1} \neq a'^{\downarrow i+1}$ or $v^{\downarrow i+1} \neq v'^{\downarrow i+1}$, either there exists a variable $z \in \mathsf{Out}_{i+1}$ such that $v^{\downarrow (i+1) \downarrow z} \neq v'^{\downarrow (i+1) \downarrow z}$ or there exists a variable $z \in \mathsf{In}_{i+1}$ such that $v^{\downarrow i+1 \downarrow z} \neq v'^{\downarrow i+1 \downarrow z}$.

  Since $\mathsf{In}_i \cup \mathsf{Out}_i \supseteq \mathsf{In}_{i+1} \cup \mathsf{Out}_{i+1}$ holds by definition of informedness, the variable z read or written on by agent $i+1$, is read or written on by agent i. Here again it follows from the construction of the game graph G_{Ar}, that either $a^{\downarrow i \downarrow z} \neq a'^{\downarrow i \downarrow z}$ or $v^{\downarrow i \downarrow z} \neq v'^{\downarrow i \downarrow z}$.

Therefore in either of the above cases, we have by definition of the game graph G_{Ar} that $a^{\downarrow i} \neq a'^{\downarrow i}$ or $v^{\downarrow i} \neq v'^{\downarrow i}$. It follows from the definition of the indistinguishability relation $\approx_i'$

that $\tau_k^\ell av \not\approx_i^\prime \tau_{k'}^\ell a'v'$, a contradiction to one of our assumptions. This completes the proof of the theorem. □

Having characterized architectures with (weak-)informedness ordering in terms of (uniform-distributed) fork-triples, the remaining part of this chapter is devoted to proving the following results:

1. The winning strategy problem for global μ-automaton winning conditions is solvable for game graphs that contain no fork-triples.
2. The deterministic winning strategy problem for global μ-automaton winning conditions is solvable for game graphs that contain no uniform-deterministic fork-triples.

We show this in two steps: first, we show that canonical (deterministic) witnesses of a game $(G, \mathscr{Q})$, where the game graph G contains no (uniform-distributed) fork-triples, belong to the class of 'modular witnesses'; second, we show that modular witnesses admit retractions to bounded size witnesses.

5.2 Modular Witnesses

Towards defining modular witnesses, we introduce some terminology associated with MaLTS's. Consider a MaLTS $\mathscr{S} = (S, R, \{\sim_i\}_{i\in[n]}, \{\nu_j\}_{j\in J}, s_\varepsilon)$.

We say that a MaLTS $\mathscr{S}$ is a *layered acyclic* MaLTS, if

1. $\mathscr{S}$ has no cycles, and all states are reachable from the start state s_ε,
2. for every state $s \in S$, every path that begins at the start state and ends at s is of the same length; we call this length, the *depth* of the state,
3. for any states $s, s' \in S$ and agent $i \in [n]$, if $s \sim_i s'$, then the states s and s' are of the same depth.

For the remainder of this section, we assume that $\mathscr{S}$ is a *layered acyclic* MaLTS.

Intuitively, a layered acyclic MaLTS may be viewed as a transition system with its states arranged in levels according to their depth, with transitions only across consecutive levels, and with indistinguishability relations contained in a level. Note that any canonical witness for an imperfect information game is a layered acyclic witness.

Modular Transition Systems An equivalence relation $\equiv \subseteq S \times S$ on the MaLTS $\mathscr{S}$ is called a *modularization*, if all states in each equivalence class in $S/\equiv$ are of the same depth. We say that a modularization $\equiv$ of the MaLTS $\mathscr{S}$,

- is *initial*, if $\equiv = \{(s,s) \mid s \in S\}$, and is *final* if $\equiv \supseteq \sim_i$ for each $i \in [n]$. Moreover we qualify the finest final modularization as *sharp*.
- is *w-wide*, if the size of every equivalence class in $S/\equiv$ is at most w.

The *modular transition system* associated with a modularization $\equiv$ of the MaLTS $\mathscr{S}$ is given by a transition system $\mathfrak{M}_\equiv = (\mathbb{M}_\equiv, \mathbb{R}_\equiv, \mathscr{M}_\varepsilon)$, where

- $\mathbb{M}_\equiv$ is the set of all substructures of $\mathscr{S}$ induced by equivalence classes in $S' \in S/\equiv$ and $\mathscr{M}_\varepsilon$ denotes the substructure induced by the state-set $[s_\varepsilon]_\equiv$, and
- $\mathbb{R}_\equiv \subseteq \mathbb{M}_\equiv \times 2^R \times \mathbb{M}_\equiv$ is the transition relation defined as follows: $(\mathscr{M}_1, R', \mathscr{M}_2) \in \mathbb{R}_\equiv$, if and only if, $R' = R \cap \mathsf{dom}(\mathscr{M}_1) \times \mathsf{dom}(\mathscr{M}_2)$ and $R' \neq \emptyset$, where $\mathsf{dom}(\mathscr{M})$ denotes the domain underlying the structure $\mathscr{M}$.

We call the states, transitions and paths of the transition system $\mathfrak{M}_\equiv$ as $\equiv$*-modular-states*, $\equiv$*-modular-transitions* and $\equiv$*-modular-paths* respectively. If the modularization being discussed is clear from the context, then we skip the parameter '$\equiv$' from each of the above terms.

Note that the states in an equivalence class induced by a modularization are of the same depth and since there are no transitions between states at the same depth by virtue of being layered acyclic, it follows that the transition relation of any modular-state is empty.

Modular Decomposition of an MaLTS For a modularization $\equiv$ of the MaLTS $\mathscr{S}$, the $\equiv$-*modular-neighbourhood* of a state $s \in S$, denoted by $\mathscr{N}_{\equiv}(s)$, is an n-tuple that satisfies the following: for each agent $i \in [n]$, if $[s]_{\sim_i} \backslash [s]_{\equiv} \neq \emptyset$, then $\mathscr{N}_{\equiv}(s)^{\downarrow i} = [s]_{\sim_i}$, otherwise $\mathscr{N}_{\equiv}(s)^{\downarrow i} = \emptyset$.

A (w,m,d)-*decomposition* of an MaLTS $\mathscr{S}$, is given by a sequence of modularizations $\equiv_1 \subseteq \equiv_2 \subseteq \ldots \subseteq \equiv_d$ of $\mathscr{S}$, that additionally satisfy the following:

1. $\equiv_1$ is a w-wide modularization and $\equiv_d$ is a final modularization.
2. the modular transition system $\mathfrak{M}_{\equiv_1}$, associated with $\equiv_1$, is a tree.
3. for any index $1 \leq c < d$ and equivalence class $S' \in S/\equiv_{c+1}$, there are at most m states in S' that have distinct $\equiv_c$-modular-neighbourhoods, that is, $|\{\mathscr{N}_{\equiv_c}(s) \mid s \in S'\}| \leq m$.

Additionally, we qualify the decomposition as *sharp*, if the final modularization $\equiv_d$ is sharp. We qualify a MaLTS as *modular*, if it admits a (w,m,d)-decomposition for some w,m,d.

Observe that for a decomposition $\equiv_1 \subseteq \equiv_2 \subseteq \ldots \subseteq \equiv_d$ of $\mathscr{S}$, if $\mathfrak{M}_{\equiv_1}$ is a tree, then every $\mathfrak{M}_{\equiv_c}$ with $c \in [d]$ is also a tree.

Next we show that (deterministic) canonical witnesses for games that have game graphs with no (uniform-deterministic) fork-triples, admit a $(1,1,n+1)$-decomposition, and are therefore modular witnesses.

Theorem 5.2.1. *Let Ar be an architecture that admits a (weak-)informedness ordering, and let G be a game graph that corresponds to the architecture Ar. Then every (deterministic) canonical witness of $(G,\mathscr{Q})$ admits a sharp $(1,1,n+1)$-decomposition.*

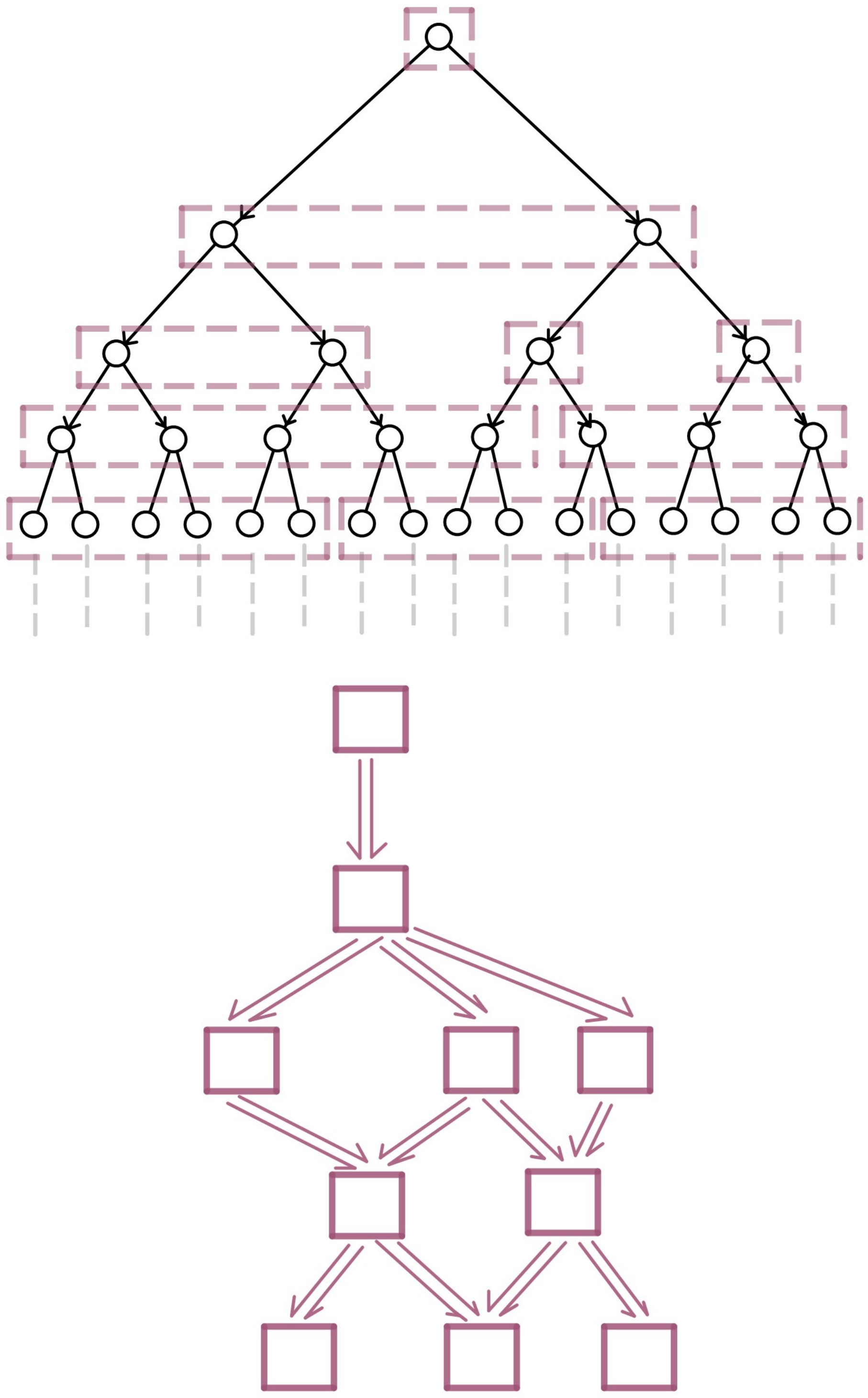


Figure 5.2: A modularization and the modular transition system.

The tree above represents a MaLTS *and the boxes drawn on the tree denote the equivalence classes induced by a modularization. The modular transition system corresponding to this modularization is drawn below, with the nodes denoting the modular-states and the edges denoting the modular-transitions. We discard the labels on the nodes and transitions for the sake of clarity.*

Proof. We simultaneously argue the claim for both the cases of uniform-deterministic fork-triples and fork-triples.

Let $\mathscr{W} = (H^\sigma, M^\sigma, \{\approx_i^\sigma\}_{i\in[n]}, \{\ell_V, \ell_Q\}, v_\varepsilon)$ be a canonical (deterministic) witness associated with a winning (deterministic) joint strategy σ, and let $\mathscr{H} = \bigcup_{i\in[n]} H^\sigma / \approx_i^\sigma$. Note that since $\mathscr{H}$ is a union of n partitionings of the set of histories H^σ, it follows that there are at most n sets in $\mathscr{H}$ that contain a state $s \in S$, and every state in S is contained in some set in $\mathscr{H}$.

Lemma 5.2.2. *The partial-order $(\mathscr{H}, \subseteq)$ is a tree-order, that is, $\emptyset \notin \mathscr{H}$ and for any distinct elements $t_1, t_2 \in \mathscr{H}$, either $t_1 \subset t_2$ or $t_2 \subset t_1$ or $t_1 \cap t_2 = \emptyset$.*

Proof. The fact that $\emptyset \notin \mathscr{H}$ follow immediately from the construction of $\mathscr{H}$. Therefore it suffices to show that for any histories $\tau, \tau' \in H^\sigma$ and agents $i, i' \in [n]$, either $[\tau]_{\approx_i^\sigma} \supseteq [\tau]_{\approx_{i'}^\sigma}$ or $[\tau]_{\approx_{i'}^\sigma} \supseteq [\tau]_{\approx_i^\sigma}$ or $[\tau]_{\approx_i^\sigma} \cap [\tau]_{\approx_{i'}^\sigma} = \emptyset$.

Towards this, consider histories $\tau, \tau' \in H^\sigma$, and assume for a contradiction that there exist agents $i, i' \in [n]$ such that, neither $[\tau]_{\approx_i^\sigma} \supseteq [\tau]_{\approx_{i'}^\sigma}$ nor $[\tau]_{\approx_{i'}^\sigma} \supseteq [\tau]_{\approx_i^\sigma}$ nor $[\tau]_{\approx_i^\sigma} \cap [\tau]_{\approx_{i'}^\sigma} = \emptyset$. Or equivalently, that $[\tau]_{\approx_i^\sigma} \setminus [\tau']_{\approx_{i'}^\sigma} \neq \emptyset$ and $[\tau']_{\approx_{i'}^\sigma} \setminus [\tau]_{\approx_i^\sigma} \neq \emptyset$. It follows from this that, there exist histories τ_1, τ_2, τ_3 such that $\tau_1 \approx_i^\sigma \tau_2 \approx_{i'}^\sigma \tau_3$ and $\tau_1 \not\approx_{i'}^\sigma \tau_2 \not\approx_i^\sigma \tau_3$.

A (uniform-deterministic) fork-triple can now be constructed using the substructure of the (deterministic) extended strategy tree $\mathscr{T}_G^\sigma$ induced by the set of prefixes of the histories τ_1, τ_2, τ_3. This is a contradiction to the assumption that the game graph G contains no (uniform-deterministic) fork-triples. The above construction of (uniform-deterministic) fork-triples crucially depends on the following observation: While extended strategy trees are guaranteed to be uniform, their substructures need not be uniform. However, when an extended strategy tree is also deterministic, then its substructures are guaranteed to be uniform and deterministic. These observations are straightforward consequences of the definition of extended strategy trees. □

We next define a *rank* on the tree-order $(\mathscr{H}, \subseteq)$ as follows: The maximal sets in $\mathscr{H}$ receive a rank n, and for a set with rank k, its children in the tree-order receive a rank $k-1$. Note that the rank is always greater than 0 for any set in $\mathscr{H}$, otherwise it would imply the existence of a sequence $H_n \subset H_{n-1} \subset .. \subset H_0$ of non-empty sets in $\mathscr{H}$, a contradiction to the fact that every history in H^σ, in particular those in H_0, is contained in at most n sets in $\mathscr{H}$.

Now consider a sequence of modularizations $\equiv_1, ..., \equiv_n$ defined as follows: Each $\equiv_k$ is such that for every $H' \in H^\sigma / \equiv_k$, either H' is a rank k set in $\mathscr{H}$ or H' is a singleton set.

Now let $\equiv_0$ be the initial modularization. It is easy to see from the above definitions and observations that $\equiv_0 \subseteq \equiv_1 \subseteq .. \subseteq \equiv_n$ holds. We claim that this sequence is a $(1,1,n+1)$-decomposition of $\mathscr{W}$, and this follows immediately from the observations below:

- $\equiv_0$ being initial, is a 1-wide modularization, and the modular transition system $\mathfrak{M}_{\equiv_0}$ is a tree that is isomorphic to the transition system underlying $\mathscr{W}$.

- For $\equiv_n$ to be a final modularization, it suffices to show that every equivalence class in $\bigcup\limits_{i \in [n]} H^\sigma / \approx_i^\sigma$ (or equivalently, $\mathscr{H}$), is a subset of some set in $H^\sigma / \equiv_n$. To see this, observe that the equivalence classes in $H^\sigma / \equiv_n$ are exactly the rank n sets in $\mathscr{H}$, and therefore contain all maximal sets in $\mathscr{H}$. Moreover since the maximal sets in $\mathscr{H}$ are equivalence classes in some $H^\sigma / \approx_i^\sigma$, it follows that $\equiv_n$ is also sharp.

- Lastly, we consider an arbitrary modularization $\equiv_k$ for some $k < n$, and show that for any two histories τ_1, τ_2 that satisfy $[\tau_1]_{\equiv_k} = [\tau_2]_{\equiv_k}$, their same $\equiv_{k-1}$-modular neighbourhood is identical, or equivalently that, for each $i \in [n]$, if $[\tau_1]_{\approx_i^\sigma} \setminus [\tau_1]_{\equiv_{k-1}} \neq \emptyset$, then $[\tau_2]_{\approx_i^\sigma} \setminus [\tau_2]_{\equiv_{k-1}} \neq \emptyset$ and $[\tau_1]_{\approx_i^\sigma} = [\tau_2]_{\approx_i^\sigma}$.

  To see this, firstly note that if $[\tau_1]_{\approx_i^\sigma} \setminus [\tau_1]_{\equiv_{k-1}} \neq \emptyset$, then from the construction of $\equiv_{k-1}$ and $\mathscr{H}$ it follows that $[\tau_1]_{\approx_i^\sigma} \supseteq [\tau_1]_{\equiv_{k-1}}$, and therefore the set $[\tau_1]_{\approx_i^\sigma}$ has rank greater than $k-1$. By construction of modularizations $\equiv_1, .. \equiv_n$, it follows that $[\tau_1]_{\approx_i^\sigma} = [\tau_1]_{\equiv_{k'}}$ for some $k' \geq k$, and therefore $[\tau_1]_{\approx_i^\sigma} \supseteq [\tau_1]_{\equiv_k}$. Since $[\tau_1]_{\equiv_k} = [\tau_2]_{\equiv_k}$, it follows that $[\tau_1]_{\approx_i^\sigma} \supseteq [\tau_2]_{\equiv_k}$ and that $[\tau_1]_{\approx_i^\sigma} = [\tau_2]_{\approx_i^\sigma}$. Moreover, since $[\tau_2]_{\equiv_k} \supseteq [\tau_2]_{\equiv_{k-1}}$ holds by construction of $\equiv_k, \equiv_{k-1}$, we have that $[\tau_1]_{\approx_i^\sigma} \supseteq [\tau_2]_{\equiv_{k-1}}$ and that $[\tau_1]_{\approx_i^\sigma} = [\tau_2]_{\approx_i^\sigma}$. The desired claim follows from this.

$\square$

5.3 Towards Retractions for Modular Witnesses

In this section we show that modular witnesses admit retractions that transform the witness into a bounded-size witness. We informally refer to such retractions as *good retractions*.

For further analysis, let us fix a layered-acyclic witness $\mathscr{W} = (S, R, \{\sim_i\}_{i \in [n]}, \{\ell_V, \ell_Q\}, s_\varepsilon)$ of the imperfect information game $(G, \mathscr{Q})$.

In order to better understand the ideas involved in this construction, we construct retractions of special cases of modular witness, and incrementally build the necessary tools to construct retractions of general modular witness. We define these special cases next.

We say that the witness $\mathscr{W}$

- has *perfect information*, if it has trivial indistinguishability relations, that is, if $\sim_i = \{(s,s) \mid s \in S\}$ for all agents $i \in [n]$.
- is *w-simple*, if there exists a modularization $\equiv$ of $\mathscr{W}$ that is final and w-wide, and if the modular transition system $\mathfrak{M}_\equiv$ associated with the modularization $\equiv$ is a tree.

5.3.1 Retraction for Perfect-Information Witnesses

Recall that a function g is a retraction of the witness $\mathscr{W}$, if it satisfies the basic constraints and the parity constraints. Note that if the witness $\mathscr{W}$ has perfect information, then the basic constraints are trivially true for any map $g : S \to S$ that preserve the labellings ℓ_V, ℓ_Q, and therefore the difficulty in constructing retractions is only in ensuring the parity constraints.

Towards fulfilling the parity constraints, we use the ideas in [33] which involve the use of 'parity progress measures'. The retraction approach in this case is conceptually identical to this solution of [33].

Parity Progress Measures The notion of 'parity progress measure' was introduced by Klarlund in [34], to give an alternative characterization of the parity condition. This has been used in many forms in the literature, and in particular, in the study of perfect information games to obtain 'small' strategies.

We begin with an informal description of the idea of a progress measure. Towards this, consider an LTS $\mathscr{X} = (X, Y, \{\ell\}, x_\varepsilon)$ with a priority labelling $\ell : X \to I$ on its states, to a set of priorities $I = \{1, 2, .., |I|\}$. Assume that $\mathscr{X}$ satisfies the *parity condition*, that is, the priority sequence $\ell(x_0), \ell(x_1), ..$ corresponding to any path $x_0 a_1 x_1 a_2 ..$ in $\mathscr{X}$ satisfies the parity condition. The parity progress measure offers a way to interpret the parity condition on infinite paths, as a 'local' property of transitions satisfying some conditions. Its usefulness in constructing retractions comes from the fact that, when one identifies a pair of states for the purpose of retracting one to the other, the local properties are easier to fulfil.

Towards formally defining these, we introduce some definitions. Let Ord denote the class of all ordinals, and let $<_{\mathsf{Ord}}$ denote the natural total order on this class. An *i-lexicographic order* on the set $\mathsf{Ord}^{|I|}$, is a binary relation $\prec_i \subseteq \mathsf{Ord}^{|I|} \times \mathsf{Ord}^{|I|}$ defined as follows: $x \prec_i x'$, if and only if, $x^{\downarrow 1} <_{\mathsf{Ord}} x'^{\downarrow 1}$ or there is a $j \leq i$ such that $x^{\downarrow j} =_{\mathsf{Ord}} x^{\downarrow j}$ for all $j < i$ and $x^{\downarrow i} <_{\mathsf{Ord}} x'^{\downarrow i}$. We note here that an i-lexicographic order is a well-founded total order, that is, it has no infinite descending chains.

A *parity progress measure* on $\mathscr{X}$ is a function $\mu : X \to \mathsf{Ord}^{|I|}$ such that, for any transition $(x, a, x') \in Y$:

1. if the priority $\ell(x)$ is even, then $\mu(x) \succeq_{\ell(x)} \mu(x')$, and

2. if the priority $\ell(x)$ is odd, then $\mu(x) \succ_{\ell(x)} \mu(x')$.

The following theorem shows the characterization of a priority-labelled LTS that satisfies the parity condition in terms of the parity progress measure.

Theorem 5.3.1 ([33])**.** *There exists a parity progress measure on $\mathscr{X}$, if and only if, the priority sequence $\ell(x_0), \ell(x_1), ..$ corresponding to any path $x_0 a_0 x_1 a_1 ..$ in $\mathscr{X}$ satisfies the parity condition.*

Parity Progress Measures for Witnesses A *parity progress measure* on the witness $\mathscr{W}$ is defined as the parity progress measure on structure $(S, R, \ell_Q \circ \Omega, s_\varepsilon)$, where Ω is the priority labelling on the states Q of the μ-automaton $\mathscr{Q}$.

Note that for the witness $\mathscr{W}$, the priority sequence $\Omega(\ell_Q(s_0)), \Omega(\ell_Q(s_1)), ..$ corresponding to any path $s_0 a_0 s_1 a_1 ..$ in $\mathscr{W}$ satisfies the parity condition. It follows by Theorem 2.3.1, there exists a parity progress measure on $\mathscr{W}$. It also follows that parity progress measures exist for any witness for the game $(G, \mathscr{Q})$.

The next theorem shows how the parity constraints can be fulfilled by means of the parity progress measure.

Theorem 5.3.2. *Let g be a function and let μ be a parity progress measure on the witness $\mathscr{W}$. If $\mu(s) \succeq_{\Omega(\ell_Q(s))} \mu(g(s))$ for all states $s \in S$, then for any g-path $s_0 a_0 s_1 a_1 ..$, the priority sequence $\Omega(\ell_Q(s_0)), \Omega(\ell_Q(s_1)), ..$ satisfies the parity condition.*

Proof. In order to see this, assume for a contradiction that $\mu(s) \succeq_{\Omega(\ell_Q(s))} \mu(g(s))$ for all states $s \in S$, and that there exists a g-path $s_0 a_0 s_1 a_1 ..$ for which the priority sequence $\Omega(\ell_Q(s_0)), \Omega(\ell_Q(s_1)), ..$ does not satisfy the parity condition. It follows from the definition of the parity condition that there exists an odd priority $p \in P$ and a suffix $s_k a_k s_{k+1} a_{k+1} ..$ of the path $s_0 a_0 s_1 a_1 ..$ such that, all priorities in the sequence $\Omega(\ell_Q(s_k)), \Omega(\ell_Q(s_{k+1})), ..$ are greater than or equal to p, and there exist infinitely many indices in this sequence where the priority is p.

Consider the sequence of states $s_0 a_0 s'_0 s_1 a_1 s'_1 ..$ such that $s'_j \in s_j R_{a_j}$ and $g(s'_j) = s_{j+1}$ for all indices j. The existence of such a sequence follows from the definition of a g-path. Now consider the priorities sequence $\Omega(\ell_Q(s_k))$,$\Omega(\ell_Q(s'_k))$,$\Omega(\ell_Q(s_{k+1}))$,$\Omega(\ell_Q(s'_{k+1})), ..$, and observe that each priority in this sequence is greater than or equal to p. This follows from our earlier observation and the first condition from the definition of a retraction, which says that $\ell_Q(s'_j) = \ell_Q(g(s'_j))$ for all $j \geq k$.

Now consider the sequence $\mu(s_k), \mu(s'_k), \mu(s_{k+1}), \mu(s'_{k+1}), ..$, and observe that $\mu(s_k) \succeq_p \mu(s'_k) \succeq_p \mu(s_{k+1}) \succeq_p \mu(s'_{k+1}) \succeq_p ..$ holds, and additionally that, whenever $\Omega(\ell_Q(s_j) = p$ is odd for some $j \geq k$, we have $\mu(s_j) \succ_p \mu(s'_j)$. This is an easy consequence of the definition of μ-progress measure, our assumption above, and the fact that every priority in the suffix $\Omega(\ell_Q(s_k), \Omega(\ell_Q(s_{k+1}), ..$ is greater than or equal to p.

Since there are infinitely many states in this suffix that have the odd priority p, it follows that the sequence $\mu(s_k) \succeq_p \mu(s'_k) \succeq_p \mu(s_{k+1}) \succeq_p \mu(s'_{k+1}) \succeq_p ..$ is an infinite descending chain in the well-founded order $\succ_p$, a contradiction to the fact that $\succ_p$ is well-founded. □

Next we show that good retractions can be constructed for witnesses that have perfect information.

Theorem 5.3.3. *If the witness $\mathscr{W}$ has perfect information, then there exists a retraction g of $\mathscr{W}$, such that $|g(\mathscr{W})| = \mathscr{O}(|V|^2.|Q|^2.|A|)$.*

Proof. Let μ be a parity-progress measure on the witness $\mathscr{W}$, and let $\leq_S \subseteq S \times S$ be a *preference-order* on the states of $\mathscr{W}$ defined as follows:

$s_1 \leq_S s_2$, if and only if, $\ell_V(s_1) = \ell_V(s_2)$, $\ell_Q(s_1) = \ell_Q(s_2)$ and $\mu(s_1) \succeq_{\Omega(\ell_Q(s_1))} \mu(s_2)$.

Note that since $\succeq_p$ is well-founded for each priority p, there are no infinite ascending chains $s_1 <_S s_2 <_S ..$ in the preference-order.

Let $f : S \to S$ be an *optimal map* on $\leq_S$, that is,

1. for any state $s \in S$, $s \leq_S f(s)$, and

2. for any states $s_1, s_2 \in f(S)$, s_1, s_2 are *incomparable* under $\leq_S$.

It is clear that since there are no infinite ascending chains in the preference-order $\leq_S$, an optimal map exists. And so, consider an optimal map g on $\leq_S$.

We claim that the function g is a retraction. To see this, note that for each element $s \in S$, $\mu(s) \leq_S \mu(g(s))$ holds by definition of an optimal map, and therefore it follows that $\ell_V(s) = \ell_V(g(s))$, $\ell_Q(s) = \ell_Q(g(s))$ and $\mu(s) \succeq_{\Omega(\ell_Q(s))} \mu(g(s))$. We know from an earlier observation that any function on S that preserves the labels ℓ_V, ℓ_Q trivially satisfies the basic constraints for being a retraction. Moreover, since $\mu(s) \succeq_{\Omega(\ell_Q(s))} \mu(g(s))$ holds for all states $s \in S$, it follows from Theorem 2.3.2 that g satisfies the parity constraints to be a retraction, and the claim follows.

To obtain the size bound for the g-retract $g(\mathscr{W})$, observe that by Proposition **??**, the domain S^g of the g-retract $g(\mathscr{W})$ is contained in $g(S)$. Now note that since each $\succeq_p$ is a total order for any priority p, any pair of states s_1, s_2 that satisfy $\ell_V(s_1) = \ell_V(s_2)$, $\ell_Q(s_1) = \ell_Q(s_2)$ are comparable under the $\leq_S$. It follows from this that any maximal set of mutually incomparable states is of size at most $|V \times Q|$. Therefore we have from the definition of an optimal map that $|g(S)| \leq |V \times Q|$. Now considering the fact that the g-retract $g(\mathscr{W})$ contains trivial indistinguishability relations, it follows that $|g(\mathscr{W})| = \mathscr{O}(|V|^2.|Q|^2.|A|)$. $\square$

Note that in the above theorem the number of priorities and agents does not contribute to the size of the resulting witness.

Corollary 5.3.4. *The winning strategy problem for perfect information games is solvable in* NPTIME.

Proof. To see this, observe that if the game $(G,\mathscr{Q})$ being considered, is a perfect information game, then the canonical witnesses of this game are perfect information witnesses. By Theorem 2.3.3, every perfect information witness of a game $(G,\mathscr{Q})$ admits a retraction into a witness of size at most $\mathscr{O}(|V|^2.|Q|^2.|A|)$. Therefore it follows from the solvability Theorem **??** that the winning strategy problem is solvable in $\text{NT}\textsc{ime}(|V|^2.|Q|^2.|A|)$, or equivalently, in $\text{NPT}\textsc{ime}$. $\square$

5.3.2 Retractions for Simple Witnesses

The retractions that we construct for simple witnesses are similar to the case of perfect information witnesses, but with the retraction mapping the domain of each modular-state as a whole, to the domain of another modular state. Because of this, it is convenient to view a retraction as a map from modular-states to other modular states. We define some terminology and formalize these ideas next.

Modular-Maps and Unbundling Consider a modularization $\equiv$ of the witness $\mathscr{W}$ and the associated modular transition system $\mathfrak{M}_{\equiv} = (\mathbb{M}_{\equiv}, \mathbb{R}_{\equiv}, \mathscr{M}_{\varepsilon})$.

- We call a function $\mathfrak{g} : \mathbb{M}_{\equiv} \to \mathbb{M}_{\equiv}$, a $\equiv$-*modular-map* on $\mathscr{W}$, if for every modular-state $\mathscr{M} \in \mathbb{M}_{\equiv}$, there exists an isomorphism from $\mathscr{M}$ to $\mathfrak{g}(\mathscr{M})$.
- A function $g : S \to S$ is called an *unbundling* of the a $\equiv$-modular map $\mathfrak{g} : \mathbb{M}_{\equiv} \to \mathbb{M}_{\equiv}$, if for every modular-state $\mathscr{M} \in \mathbb{M}_{\equiv}$, the restriction of the function g to the domain $\mathsf{dom}(\mathscr{M})$, is an isomorphism from $\mathscr{M}$ to $\mathfrak{g}(\mathscr{M})$.

Theorem 5.3.5. *Let g be an unbundling of some $\equiv$-modular map $\mathfrak{g}$ on $\mathscr{W}$. Then,*

1. *for any state $s \in S$, $\ell_V(s) = \ell_V(g(s))$ and $\ell_Q(s) = \ell_Q(g(s))$,*
2. *for any states $s_1, s_2 \in S$ and agent $i \in [n]$, if $s_1 \equiv s_2$ holds, then $s_1 \sim_i s_2$, if and only if, $g(s_1) \sim_i g(s_2)$, and*

3. $g(S) = \bigcup_{\mathscr{M} \in \mathfrak{g}(\mathbb{M}_{\equiv})} \mathsf{dom}(\mathscr{M})$.

4. *g is a retraction of the witness $\mathscr{W}$, if the following conditions hold:*

   (a) for any states $s_1, s_2 \in S$ and agent $i \in [n]$, if $s_1 \not\equiv s_2$, then $s_1 \sim_i s_2$, if and only if, $g(s_1) \sim_i g(s_2)$.

   (b) for any g-path $s_0 a_0 s_1 a_1 \ldots$ that begins at the state $g(s_\varepsilon)$, the priority sequence $\Omega(\ell_Q(s_0)), \Omega(\ell_Q(s_1)), \Omega(\ell_Q(s_2)), ..$ satisfies the parity condition.

Proof. The first, second and third items in the above theorem are immediate consequences of the definition of $\equiv$-modular maps and unbundling.

For the proof of the fourth item, consider the following observations:

- For g to be a retraction $\ell_V(s) = \ell_V(g(s))$ and $\ell_Q(s) = \ell_Q(g(s))$ must hold for any state $s \in S$, and this follows from item 1. The parity constraint for g to be a retraction is verbatim the second condition in the theorem.

- It remains to show that for any agent $i \in [n]$ and transitions $(s_1, a, s_2), (s'_1, a', s'_2) \in R^g$, if $s_1 \sim_i s_2$, then, $s_2 \sim_i s'_2$, if and only if, $\ell_V(s'_1)^{\downarrow i} = \ell_V(s'_2)^{\downarrow i}$ and $a^{\downarrow i} = a'^{\downarrow i}$.

  Towards this consider an agent $i \in [n]$ and transitions $(s_1, a, s_2), (s'_1, a', s'_2) \in R^g$ such that $s_1 \sim_i s'_1$. Then by definition of R^g, there must exist states $s, s' \in S$ and transitions $(s_1, a, s), (s'_1, a', s') \in R$ such that $s_2 = g(s)$ and $s'_2 = g(s')$ hold. The desired conclusion follows immediately from the following two claims.

  – Firstly we claim that $s_2 \sim_i s'_2$, if and only if, $s \sim_i s'$. To see this, observe that if $s \equiv s'$, then the claim follow immediately from item 2. On the other hand if $s \not\equiv s'$, then the claim follows from the first condition in the theorem.

  – Secondly we claim that $s \sim_i s'$, if and only if, $\ell_V(s)^{\downarrow i} = \ell_V(s')^{\downarrow i}$ and $a^{\downarrow i} = a'^{\downarrow i}$. To see this, consider the Indistinguishability condition (that is, condition **V.2**) satisfied by the witness $\mathscr{W}$, stated below:

$\forall i \in [n], \forall (s_1, a, s_2), (s'_1, a', s'_2) \in R:$ if $s_1 \sim_i s'_1$, then,
$s_2 \sim_i s'_2$, if and only if, $\ell_V(s_2)^{\downarrow i} = \ell_V(s'_2)^{\downarrow i}$ and $a^{\downarrow i} = a'^{\downarrow i}$.

Since $s_1 \sim_i s'_1$ holds and transitions $(s_1, a, s), (s'_1, a', s') \in R$ exist by our earlier observation, it follows from the above condition that, $s \sim_i s'$, if and only if, $\ell_V(s)^{\downarrow i} = \ell_V(s')^{\downarrow i}$ and $a^{\downarrow i} = a'^{\downarrow i}$.

This completes the proof of the theorem. $\square$

Analogous to the case of retraction for the perfect-information case, the construction of retractions of the w-simple case proceeds by considering a 'preference-order' on modular-states and 'optimal maps' on preference-order. We define these next.

Modular-Preference-Order and Optimal Maps Consider a modularization $\equiv$ of the witness $\mathscr{W}$ and the associated modular transition system $\mathfrak{M}_{\equiv} = (\mathbb{M}_{\equiv}, \mathbb{R}_{\equiv}, \mathscr{M}_{\varepsilon})$.

- A pre-order $\trianglelefteq \subseteq \mathbb{M}_{\equiv} \times \mathbb{M}_{\equiv}$ is called a $\equiv$-*modular-preference-order*, if it satisfies the following: for all $\mathscr{M}_1, \mathscr{M}_2 \in \mathbb{M}_{\equiv}$, if $\mathscr{M}_1 \trianglelefteq \mathscr{M}_2$, then there exists an isomorphism from $\mathscr{M}_1$ to $\mathscr{M}_2$.
- Given a modular-preference-order $\trianglelefteq$, we qualify a modular-map $\mathfrak{g}: \mathbb{M}_{\equiv} \to \mathbb{M}_{\equiv}$
  - as $\trianglelefteq$-*monotone*, if $\mathscr{M} \trianglelefteq \mathfrak{g}(\mathscr{M})$ for all modular-states $\mathscr{M} \in \mathbb{M}_{\equiv}$.
  - as $\trianglelefteq$-*optimal*, if it is $\trianglelefteq$-monotone, every modular-state in $\mathfrak{g}(\mathbb{M}_{\equiv})$ is a maximal modular-state under the ordering $\trianglelefteq$, and no two modular-states in $\mathfrak{g}(\mathbb{M}_{\equiv})$ are comparable under $\trianglelefteq$.

Note that an $\trianglelefteq$-optimal modular-map may not exist for any modular-preference-order $\trianglelefteq$. A necessary and sufficient criterion for the existence of this is given next.

Proposition 5.3.6. *For any modular-preference-order $\trianglelefteq$, there exists a $\trianglelefteq$-optimal modular map, if and only if, there are no infinite ascending chains in $\trianglelefteq$.*

We now return to our first construction of retraction for the case where the witness $\mathscr{W}$ is w-simple. We show that in this case, the parity progress measure is inadequate for the purpose of constructing good retractions.

Theorem 5.3.7. *If the witness $\mathscr{W}$ is w-simple, then there exists a retraction g of $\mathscr{W}$ such that, the g-retract $g(\mathscr{W})$ is finite.*

Proof. Let $\mathscr{W}$ have $\equiv$ as its final and w-wide modularization. Let $\mathfrak{M}_{\equiv} = (\mathbb{M}_{\equiv}, \mathbb{R}_{\equiv}, \mathscr{M}_{\varepsilon})$ denote the associated modular transition system and let μ be the parity progress measure of the witness $\mathscr{W}$.

Now consider the modular-preference-order $\trianglelefteq \subseteq \mathbb{M}_{\equiv} \times \mathbb{M}_{\equiv}$ defined as follows:
$\mathscr{M}_1 \trianglelefteq \mathscr{M}_2$, if and only if, there is an isomorphism h from $\mathscr{M}_1$ to $\mathscr{M}_2$ such that, $\mu(s) \succeq_{\Omega(\ell_Q(s))} \mu(h(s))$ for all states $s \in \mathsf{dom}(\mathscr{M}_1)$.

Firstly note that there are no infinite ascending chains in $\trianglelefteq$. To see this, consider for a contradiction an infinite ascending chain $\mathscr{M}_1 \triangleleft \mathscr{M}_2 \triangleleft ...$ By definition of $\trianglelefteq$, there exists an isomorphism $h_i : \mathsf{dom}(\mathscr{M}_i) \to \mathsf{dom}(\mathscr{M}_{i+1})$ for each index i. It is easy to see from this that there exists some state $s \in \mathsf{dom}(\mathscr{M}_1)$ such that $\mu(s) \succ_{\Omega(\ell_Q(s))} \mu(h_1(s)) \succ_{\Omega(\ell_Q(s))} \mu(h_2(h_1(s))) \succ_{\Omega(\ell_Q(s))}$ .., a contradiction to the well-foundedness of $\succ_{\Omega(\ell_Q(s))}$.

Since there are no infinite ascending chains in $\trianglelefteq$, it follows from 2.3.6 that a $\trianglelefteq$-optimal modular-map $\mathfrak{g}$ exists on $\trianglelefteq$. Let g be the unbundling of $\mathfrak{g}$.

Firstly we claim that g is a retraction of $\mathscr{W}$. To see this, consider item 4 of Theorem 2.3.5, and observe the following:

- For any states $s_1, s_2 \in S$, if $s_1 \sim_i s_2$ for some agent $i \in [n]$, then due to the modularization $\equiv$ being final, it follows that $s_1 \equiv s_2$. Therefore the first condition in item 4 of Theorem 2.3.5 holds trivially.

- By construction of g, it follows that $\mu(s) \succeq_{\Omega(\ell_Q(s))} \mu(g(s))$ for each $s \in S$, and therefore by Proposition 2.3.2, the second condition in item 4 of Theorem 2.3.5

holds.

The fact that g is a retraction follows from item 4 of Theorem 2.3.5.

It remains to show that $g(\mathscr{W})$ is a finite. Note that by Proposition **??** and Theorem 2.3.5, we have that S^g (the domain of $g(\mathscr{W})$) satisfies $S^g \subseteq \bigcup_{\mathscr{M} \in \mathfrak{g}(\mathbb{M}_\equiv)} \mathsf{dom}(\mathscr{M})$. Since every modular-state in $\mathfrak{g}(\mathbb{M}_\equiv)$ is maximal, it must the case that for any two modular states $\mathscr{M}, \mathscr{M}' \in \mathfrak{g}(\mathbb{M}_\equiv)$, either they are non-isomorphic, or there exists an isomorphism from $\mathscr{M}, \mathscr{M}'$ but the *progress measure profiles* $(\mu(s))_{s \in \mathsf{dom}(\mathscr{M})}$ and $(\mu(h(s)))_{s \in \mathsf{dom}(M)}$ are incomparable, when ordered point-wise under $\prec_{\Omega(\ell_Q(s))}$ at each index s. It is an easy consequence of Dickson's lemma that there are only finitely many progress measure profiles that are incomparable to a profile $(\mu(s))_{s \in \mathsf{dom}(\mathscr{M})}$. Since there are only finitely many modular-states that are incomparable under isomorphism (by virtue of $\equiv$ being w-wide), we can conclude that there are only finitely many modular-states in $\mathfrak{g}(\mathbb{M}_\equiv)$. Since each modular-state is a finite structure, it follows that S^g is finite, and therefore $g(\mathscr{W})$ is finite. □

As shown above, the direct use of parity progress measure is insufficient to provide good retractions in the case where $\mathscr{W}$ is w-simple, since the set $\mathfrak{g}(\mathbb{M}_\equiv)$, although of finite size, could contain arbitrarily large number of modular-states. In the remaining part of this section, we build techniques to handle this. We resolve this problem by defining a generalization of a parity progress measure called a 'modular-progress measure', that may be seen as a parity progress measures on modular transition systems.

5.3.3 Constructing Modular-Progress Measure

We begin with a summary of the construction of modular progress measure. Towards this, consider the witness $\mathscr{W}$ and a modularization $\equiv$ of $\mathscr{W}$ such that, $\equiv$ is w-wide and the associated modular transition system $\mathfrak{M}_\equiv = (\mathbb{M}_\equiv, \mathbb{R}_\equiv, \mathscr{M}_\varepsilon)$ is a tree.

The construction of a $\equiv$-modular progress measure proceeds in three steps:

1. First, we construct an alphabet $\Sigma_{\equiv}$, and provide an interpretation of modular-paths in $\mathfrak{M}_{\equiv}$ as a words over $\Sigma_{\equiv}$.

2. Second, we construct a deterministic parity ω-word automaton such that, for all modular-paths in $\mathfrak{M}_{\equiv}$, their corresponding interpretation as words over $\Sigma_{\equiv}$, is accepted by the automaton.

3. Lastly, we define the notion of modular-progress measure, and show that retractions that use this notion of progress measure are good retractions.

Interpreting Modular-paths as ω-Words Consider an arbitrary total-ordering on the set of states S of $\mathscr{W}$, and let the term i^{th} *largest element* of a set $S' \subseteq S$, refer to the i^{th} largest element of the set S' under the this total ordering.

A structure T is called a *transition structure* of the modular-transition $(\mathscr{M}_1, R_1, \mathscr{M}_2) \in \mathbb{R}_{\equiv}$, if T is a substructure of $\mathscr{W}$ induced by the set of states $\mathsf{dom}(\mathscr{M}_1) \cup \mathsf{dom}(\mathscr{M}_2)$. We denote the states in $\mathsf{dom}(\mathscr{M}_1)$ and $\mathsf{dom}(\mathscr{M}_2)$ as $\mathsf{source}(T)$ and $\mathsf{sink}(T)$ respectively.

Let $\Sigma_{\equiv}$ be a minimal set of transition structures such that for every modular-transition $(\mathscr{M}_1, R_1, \mathscr{M}_2) \in \mathbb{R}_{\equiv}$, there exists an element $T \in \Sigma_{\equiv}$ that is isomorphic to the transition structure of $(\mathscr{M}_1, R_1, \mathscr{M}_2)$. Note that since $\equiv$ is w-wide, the set $\Sigma_{\equiv}$ is finite and of size at most $\mathscr{O}(w^2)$.

For a sequence of modular-transitions $(\mathscr{M}_1, R_1, \mathscr{M}'_1), (\mathscr{M}_2, R_2, \mathscr{M}'_2), ..$, its *word interpretation* is given by the sequence $T_1, T_2, ..$, where each $T_i \in \Sigma_{\equiv}$ is isomorphic to the transition structure of the modular-transition $(\mathscr{M}_i, R_i, \mathscr{M}_{i+1})$. The *word interpretation* of a modular-path $\mathscr{M}_1, R_1, \mathscr{M}_2, R_2, ..$, is defined as the word interpretation of the sequence $(\mathscr{M}_1, R_1, \mathscr{M}_2), (\mathscr{M}_2, R_2, \mathscr{M}_3), ..$

Automaton on Modular-Paths Next we construct an automaton that accepts word-interpretations of those modular-paths for which the paths 'contained in' a modular-path, satisfy the $\mathscr{Q}$-*Parity* condition.

Towards this, recall that P is the set of priorities used to label the states of the automaton $\mathscr{Q}$, and let p_ε denote the priority $\Omega(q_\varepsilon)$ of the start state q_ε.

A *thread* in a word $T_1, T_2, .. \in \Sigma_{\equiv}^{\omega}$, is any sequence $(w_1, p_1), (w_2, p_2), .. \in ([w] \times P)^{\omega}$, such that, for any index $i \geq 1$, there exists some transition (s_i, a, s_{i+1}) in T_i that satisfies the following:

1. s_i is the w_i^{th} largest element in $\mathsf{source}(T_i)$ and $\Omega(\ell_Q(s_i)) = p_i$.

2. s_{i+1} is the w_{i+1}^{th} largest element in $\mathsf{sink}(T_i)$ and $\Omega(\ell_Q(s_{i+1})) = p_{i+1}$.

We say that an ω-word $T_1, T_2, .. \in \Sigma_{\equiv}^{\omega}$ satisfies the *parity condition*, if and only if, for every thread $(w_1, p_1), (w_2, p_2), ..$ of the ω-word $T_1, T_2, ..$ that begins at $(1, p_\varepsilon)$, the priority sequence $p_1, p_2, ..$ satisfies the parity condition.

Let L denote the set of ω-words in $\Sigma_{\equiv}^{\omega}$ that satisfy the parity condition. Note that since word interpretations of modular-paths satisfy the parity condition, they belong to the language L. The next proposition shows that the language L is ω-regular.

Proposition 5.3.8. *There exists a deterministic parity ω-word automaton that accepts L and has $2^{\mathcal{O}(|P|.w)}$ states.*

Proof. The construction of the desired automaton proceeds in three steps. First, we construct a parity automaton $\mathscr{A}'$ that accepts the language $\overline{L}$. Second, we convert it into a Büchi automaton and use Theorem **??** stated in Chapter 2 to obtain an equivalent deterministic parity automaton $\mathscr{A}''$. And third, we obtain a deterministic parity automaton $\mathscr{A}'''$ that is the complement of $\mathscr{A}''$.

As the first step, let $\mathscr{A}' = (Q', \Sigma_{\equiv}, \Delta', \Omega', (1, \Omega(q_\varepsilon)))$ be a parity automaton, where

- $Q' = ([w] \times |P|) \cup \{q'_\perp\}$ with $(1, p_\varepsilon)$ as the start state.
- for any transition structure $T \in \Sigma_\equiv$,
  - $(q'_\perp, T, q'_\perp) \in \Delta'$,
  - $((1, p_\varepsilon), T, q'_\perp) \in \Delta'$, if and only if, the smallest element in $\mathsf{source}(T)$ has a priority different from p_ε.
  - for states $(w_1, p_1), (w_2, p_2) \in Q' \setminus \{q_\perp\}$, we have $((w_1, p_1), T, (w_2, p_2)) \in \Delta'$, if and only if, there exists a transition $(s_1, a, s_2) \in R_1$ such that
    1. s_1 is the w_1^{th} largest element in $\mathsf{source}(T)$ and $\Omega(\ell_Q(s_1)) = p_1$.
    2. s_2 is the w_2^{th} largest element in $\mathsf{sink}(T)$ and $\Omega(\ell_Q(s_2)) = p_2$.
- $\Omega' : Q' \to \{1, 2, .., p+1\}$ satisfies $\Omega'((w_i, p_i)) = p_i + 1$ for any $(w_i, p_i) \in Q' \setminus \{q_\perp\}$, and $\Omega'(q_\perp) = 0$.

It is not difficult to see that, for any sequence $(w_1, p_1), (w_2, p_2), .. \in ([w] \times |P|)^\omega$ and ω-word $T_1, T_2, .. \in \Sigma_\equiv$, the sequence $(w_1, p_1)T_1(w_2, p_2)T_2..$ is a run of the automaton $\mathscr{A}'$, if and only if, $(w_1, p_1), (w_2, p_2), ..$ is a thread of the ω-word $T_1, T_2, ..$ that begins at $(1, p_\varepsilon)$.

We argue next that the automaton $\mathscr{A}'$ recognizes the language $\overline{L}$. To see that every ω-word in $\overline{L}$ is accepted by the automaton $\mathscr{A}'$, consider an ω-word $T_1, T_2, .. \in \overline{L}$. Since $T_1, T_2, .. \in \overline{L}$, one of the cases below applies:

- Either every thread $(w_1, p_1), (w_2, p_2), ..$ in $T_1, T_2, ..$ begins at an pair other than $(1, p_\varepsilon)$, in which case there is a run $(w_1, p_1)T_1 q'_\perp T_2 q'_\perp ..$ of automaton $\mathscr{A}'$, that at the start state $(1, p_\varepsilon)$ of the automaton, proceeds to the state $q'_\perp$ and remains there. The run is an accepting run, since the priority of $\Omega'(q'_\perp)$ is 0 and it follows that the priority sequence $\Omega'((w_1, p_1))\Omega'(q'_\perp)\Omega'(q'_\perp)...$ satisfies the parity condition.
- Or there exists a thread $(w_1, p_1), (w_2, p_2), ..$ in $T_1, T_2, ..$ that begins at $(1, p_\varepsilon)$, such that the priority sequence $p_1, p_2, ..$ does not satisfy the parity conditions. In this

case, the sequence $(w_1, p_1)T_1(w_2, p_2)T_2..$ is guaranteed to be a run of the automaton; the definition of Δ' is set up exactly so that this happens. This run is an accepting run because by definition of the priority labelling Ω', the priority sequence $\Omega'((w_1, p_1)), \Omega'((w_2, p_2)), ..$ is identical to $p_1+1, p_2+1, ..$, and therefore it follows that the priority sequence satisfies the parity condition.

To see that an ω-word not in $\overline{L}$ is not accepted by the automaton $\mathscr{A}'$, consider an ω-word $T_1, T_2, .. \notin \overline{L}$. Then every run $(w_1, p_1)T_1(w_2, p_2)T_2..$ of $T_1, T_2, ..$ on $\mathscr{A}'$ corresponds exactly to some thread $(w_1, p_1), (w_2, p_2)..$ of $T_1, T_2, ...$ Since $T_1, T_2, .. \notin \overline{L}$, or equivalently, $T_1, T_2, .. \in L$, it follows from the definition of L that the priority sequence $p_1, p_2, ..$ satisfies the parity condition, and therefore $(w_1, p_1)T_1(w_2, p_2)T_2..$ is not an accepting run.

Now using Theorem **??**, we convert the above parity automaton $\mathscr{A}'$, which has $w.|P|+1$ states and $|P|+1$ priorities, to a Büchi automaton that recognizes the same language as $\mathscr{A}'$, and has $k = (w.|P|+1).(|P|+1)$ states. Using Theorem **??**, we further convert this Büchi automaton to a deterministic parity automaton $\mathscr{A}''$ that recognizes the same language as $\mathscr{A}'$, and has k^{2k+2} states with $2k$-many priorities. Again using Theorem **??**, we obtain a deterministic parity automaton $\mathscr{A}'''$ that recognizes the complement language recognized by $\mathscr{A}''$, and has the same size and number of priorities as $\mathscr{A}''$. Since $k = \mathscr{O}(w.|P|^2)$, the desired bound follows. $\square$

Modular-progress measure Now let $\mathscr{A}^L = (Q^L, \Sigma_{\equiv}, \Delta^L, \Omega^L, q^L_{\varepsilon})$ be a deterministic parity automaton that recognizes the language L described earlier, and let Ω^L be a function from Q^L to the set of priorities P^L.

A $\equiv$-*modular progress measure* of the witness $\mathscr{W}$, is a tuple $(\mu_{\equiv}, \ell_{Q^L}, \ell_{P^L})$, where

- $\ell_{Q^L} : \mathbb{M}_{\equiv} \to Q^L$ is a function such that, for any modular-path $\mathscr{M}_1, R_1, \mathscr{M}_2, R_2, ..$ in the modular transition system $\mathfrak{M}_{\equiv}$ that begins at $\mathscr{M}_{\varepsilon}$, the state sequence $\ell_{Q^L}(\mathscr{M}_1)\ell_{Q^L}(\mathscr{M}_2)..$ is the unique run of the automaton $\mathscr{A}^L$ on the word-interpretation of the modular-path

$\mathscr{M}_1, R_1, \mathscr{M}_2, R_2, ..,$

- $\ell_{P^L} = \ell_{Q^L} \circ \Omega^L$ is function that maps modular-states in $\mathbb{M}_\equiv$ to the set of priorities P^L.
- $\mu_\equiv$ is a parity progress measure on the LTS $(\mathbb{M}_\equiv, \mathbb{R}_\equiv, \{\ell_{P^L}\}, \mathscr{M}_\varepsilon)$.

Note that since the automaton is deterministic and the modular transition system $\mathfrak{M}_\equiv$ is a tree, the function ℓ_{Q^L} is well-defined. Intuitively, ℓ_{Q^L} labels $\mathfrak{M}_\equiv$ in such a way that the labelling along a modular-path corresponds to a run of the automaton $\mathscr{A}^L$ on the word interpretation of the modular-path. The threads of this word interpretation correspond to priority sequences of paths 'contained' in the modular path, and since threads of words in L satisfy the parity condition, the $\mathscr{Q}$-Parity condition on paths is maintained.

Next we exhibit the usefulness of modular progress measures in the construction of good retractions.

Theorem 5.3.9. *Let $\equiv$ be a w-wide modularization of the witness $\mathscr{W}$ and let $\mathfrak{M}_\equiv = (\mathbb{M}_\equiv, \mathbb{R}_\equiv, \mathscr{M}_\varepsilon)$, the modular transition system associated with $\equiv$, be a tree.*

1. *There exists a $\equiv$-modular progress measure of the witness $\mathscr{W}$.*
2. *Let $(\mu_\equiv, \ell_{Q^L}, \ell_{P^L})$ be a $\equiv$-modular progress measure of the witness $\mathscr{W}$.*

   Let $\trianglelefteq$ be the $\equiv$-modular-preference-order such that, for any $\mathscr{M}_1, \mathscr{M}_2 \in \mathbb{M}_\equiv$, if $\mathscr{M}_1 \trianglelefteq \mathscr{M}_2$, then $\ell_{Q^L}(\mathscr{M}_1) = \ell_{Q^L}(\mathscr{M}_2)$ and $\mu_\equiv(\mathscr{M}_1) \succeq_{\ell_{P^L}(\mathscr{M}_1)} \mu_\equiv(\mathscr{M}_2)$.

   Then the following hold:

   (a) There exists a $\trianglelefteq$-optimal modular-map.

   (b) For any $\trianglelefteq$-monotone modular-map $\mathfrak{g}$, its unbundling g and any g-path $s_0 a_0 s_1 a_1 ..$, the priority sequence $\Omega(\ell_Q(s_0)), \Omega(\ell_Q(s_1)), ..$ satisfies the parity condition.

Proof. Consider the alphabet $\Sigma_\equiv$, the language L as defined earlier. Let $\mathscr{A}_L = (Q^L, \Sigma_\equiv, \Delta^L, \Omega^L, q^L_\varepsilon)$ be a deterministic parity ω-word automaton that accepts L. The existence of such an automaton follows from Proposition 2.3.8. Also, let the $\equiv$-modular progress measure $(\mu_{\equiv}, \ell_{Q^L}, \ell_{P^L})$ be obtained from this automaton $\mathscr{A}_L$ as described earlier.

The first part of the theorem is an immediate consequence of the preceding construction of the $\equiv$-modular progress measure.

For the first item of the second part, note that if there exists an infinite ascending chain $\mathscr{M}_1 \lhd \mathscr{M}_2 \lhd \mathscr{M}_3 ...$, then it follows from the definition of $\lhd$ and Φ that $\mu_{\equiv}(\mathscr{M}_1) \succ_{\ell_{pL}(\mathscr{M}_1)} \mu_{\equiv}(\mathscr{M}_2) \succ_{\ell_{pL}(\mathscr{M}_1)} \mu_{\equiv}(\mathscr{M}_3)..$, a contradiction to the well-foundedness of $\succ_{\ell_{pL}(\mathscr{M}_1)}$. Therefore it follows from Theorem 2.3.6 that an optimal modular map exists.

Towards the second item of the second part, we consider a g-path $s_0 a_1 s_1 a_2 ...$ and the following objects:

- Let $(s_0, a_0, s_0'), (s_1, a_1, s_1'), ...$ be a transition sequence such that for each index i, (s_i, a_i, s_i') is a transition in the witness $\mathscr{W}$ and $g(s_i') = s_{i+1}$. The existence of such a sequence follows from the definition of the g-path $s_0 a_1 s_1 a_2 ....$

- Let $(\mathscr{M}_0, R_0, \mathscr{M}_0'), (\mathscr{M}_1, R_1, \mathscr{M}_1'), ...$ be a sequence of modular-transitions, such that for each index i, we have $\mathfrak{g}(\mathscr{M}_i') = \mathscr{M}_{i+1}$, and $s_i \in \mathsf{dom}(\mathscr{M}_i)$, $s_i' \in \mathsf{dom}(\mathscr{M}_i')$ and $(s_i, a_i, s_i') \in R_i$. The existence of such a sequence follows from the definition of the modular transition relation $\mathbb{R}_{\equiv}$ and the construction of the optimal map g.

  Since $\mathfrak{g}$ is $\unlhd$-monotone, it follows that $\mathscr{M}_i' \unlhd \mathscr{M}_{i+1}$ for all indices i, and therefore $\ell_{Q^L}(\mathscr{M}_i') = \ell_{Q^L}(\mathscr{M}_{i+1})$ and $\mu_{\equiv}(\mathscr{M}_i') \succeq_p \mu_{\equiv}(\mathscr{M}_{i+1})$,

- Let $T_0, T_1, T_2, .. \in \Sigma_{\equiv}^{\omega}$ be the word interpretation of the modular-transition sequence $(\mathscr{M}_0, R_0, \mathscr{M}_0'), (\mathscr{M}_1, R_1, \mathscr{M}_1'), ...$ and let $(w_0, p_0), (w_1, p_1), .. \in ([w] \times P)^{\omega}$ be a sequence such that for every index i, s_i is the w_i^{th} largest element in $\mathscr{M}_i$ and has priority $\Omega(\ell_Q(s_i)) = p_i$. It follows easily that the sequence $(w_0, p_0), (w_1, p_1), ..$ is a thread in $T_0, T_1, T_2, ..$, since for every index i, the modular-transition associated with each transition structure T_i, is isomorphic to the modular-transition $(\mathscr{M}_i, R_i, \mathscr{M}_i')$.

We claim that $T_0, T_1, T_2, ..$ belongs to the language L, and that the sequence $q_0^L T_0 q_1^L T_1 q_2^L T_2 ..$ that satisfies $q_i^L = \ell_{Q^L}(\mathscr{M}_i)$ for each index i, is an accepting run of the automaton $\mathscr{A}^L$ on this word.

Now assuming that this claim is true, it follows from the definition of L that the priority sequence $p_0, p_1, ..$ associated with the thread $(w_0, p_0), (w_1, p_1), ..$ in the word $T_0, T_1, T_2, ..$, satisfies the parity condition. Since each $p_i = \Omega(\ell_Q(s_i))$, the priority sequence $\Omega(\ell_Q(s_0)), \Omega(\ell_Q(s_1)), ..$ satisfies the parity condition, and the claim follows.

Towards proving the above claim that $T_0, T_1, T_2, ..$ belongs to the language L, consider the associated modular-transition sequence $(\mathscr{M}_0, R_0, \mathscr{M}'_0), (\mathscr{M}_1, R_1, \mathscr{M}'_1), ...$, and the sequence of states $q_0^L, q_0'^L, q_1^L, q_0'^L$ such that $q_i^L = \ell_{Q^L}(\mathscr{M}_i)$ and $q_i'^L = \ell_{Q^L}(\mathscr{M}'_i)$ hold for every index i. It follows that for any index i, $\Delta^L(q_i^L, T_i) = q_i'^L$ holds by construction of the labelling ℓ_{Q^L} and $q_i'^L = q_{i+1}^L$ due to our earlier observation that $\ell_{Q^L}(\mathscr{M}'_i) = \ell_{Q^L}(\mathscr{M}_{i+1})$. Therefore the sequence $q_0^L T_0 q_1^L T_1 q_2^L T_2 ..$ is a run of the ω-word $T_0, T_1, T_2, ..$ on the automaton $\mathscr{A}^L$.

It remains to show that $q_0^L T_0 q_1^L T_1 q_2^L T_2 ..$ is also an accepting run. Towards this, consider the priority sequence $\Omega^L(q_0^L), \Omega^L(q_0'^L), \Omega^L(q_1^L), \Omega^L(q_1'^L), ..$, or equivalently, the priority sequence $\ell_{P^L}(\mathscr{M}_0), \ell_{P^L}(\mathscr{M}'_0), \ell_{P^L}(\mathscr{M}_1), \ell_{P^L}(\mathscr{M}'_1), ...$ We claim that this sequence satisfies the parity condition. Towards proving this, assume for a contradiction that the sequence does not satisfy the parity condition. By definition of parity condition, it follows that there exists an odd priority $p \in P^L$ and a suffix $\ell_{P^L}(\mathscr{M}_i), \ell_{P^L}(\mathscr{M}'_i), \ell_{P^L}(\mathscr{M}_{i+1}), \ell_{P^L}(\mathscr{M}'_{i+1}), ..$ such that, all priorities appearing in this suffix are greater than or equal to p and p appears infinitely often in this suffix. Now observe that for all indices $j \geq i$,

- $(\mathscr{M}_j, R_j, \mathscr{M}'_j)$ is a transition in $\mathbb{R}_{\equiv}$, and therefore $\mu_{\equiv}(\mathscr{M}_j) \succeq_p \mu_{\equiv}(\mathscr{M}'_j)$ holds by definition of the parity progress measure $\mu_{\equiv}$, with the additional constraint that $\mu_{\equiv}(\mathscr{M}_j) \succ_p \mu_{\equiv}(\mathscr{M}'_j)$ whenever $\ell_{P^L}(\mathscr{M}_j) = p$.
- $\ell_{P^L}(\mathscr{M}'_j) \geq p$ and $\mu_{\equiv}(\mathscr{M}'_j) \succeq_p \mu_{\equiv}(\mathscr{M}_{j+1})$ hold due to an earlier observation.

It follows that the sequence $\mu_{\equiv}(\mathscr{M}_i) \succeq_p \mu_{\equiv}(\mathscr{M}'_i) \succeq_p \mu_{\equiv}(\mathscr{M}_{i+1}) \succeq_p \mu_{\equiv}(\mathscr{M}'_{i+1}), ..$ holds, with the additional constraint that the strict order $\mu_{\equiv}(\mathscr{M}_j) \succ_p \mu_{\equiv}(\mathscr{M}'_j)$ whenever $\ell_{P^L}(\mathscr{M}_j) = p$ for all $j \geq i$. This is a contradiction to the fact that $\succ_p$ is a well-founded order. Therefore, we conclude that the sequence $q_0^L T_0 q_1^L T_1 q_2^L T_2..$ is an accepting run of the automaton $\mathscr{A}^L$. $\square$

Theorem 5.3.10. *If $\mathscr{W}$ is w-simple, then there exists a retraction g such that the g-retract $g(\mathscr{W})$ has a domain of size $2^{\mathscr{O}(n.w^2\ log(|V|.|Q|.|P|))}$.*

Proof. Let $\equiv$ denote a final and w-wide modularization of the $\mathscr{W}$. Let $(\mu_{\equiv}, \ell_{Q^L}, \ell_{P^L})$ be a $\equiv$-modular progress measure of the witness $\mathscr{W}$. The fact that this exists follows from Theorem 2.3.9.

Now consider the modular-preference-order $\trianglelefteq \subseteq \mathbb{M}_{\equiv} \times \mathbb{M}_{\equiv}$ defined as follows:
$\mathscr{M}_1 \trianglelefteq \mathscr{M}_2$, if and only if, there exists an isomorphism from $\mathscr{M}_1$ to $\mathscr{M}_2$, $\ell_{Q^L}(\mathscr{M}_1) = \ell_{Q^L}(\mathscr{M}_2)$ and $\mu_{\equiv}(\mathscr{M}_1) \succeq_{\ell_{P^L}(\mathscr{M}_1)} \mu_{\equiv}(\mathscr{M}_2)$.

According to Theorem 2.3.9, for a modular-preference-order of the form defined above, there exists a $\trianglelefteq$-optimal modular-map $\mathfrak{g}$. Let g denote the unbundling of the modular-map $\mathfrak{g}$, and consider the following:

- For any states $s_1, s_2 \in S$, if $s_1 \sim_i s_2$ for some agent $i \in [n]$, then due to the modularization $\equiv$ being final, it follows that $s_1 \equiv s_2$.
- For any g-path $s_0 a_1 s_1 a_2 ...$, the priority sequence $\Omega(\ell_Q(s_0)), \Omega(\ell_Q(s_1)),..$ satisfies the parity condition. This is again a consequence of Theorem 2.3.9.

It now follows from the above two observations and item 4 of Theorem 2.3.5 that g is a retraction of the witness $\mathscr{W}$.

For the size bound on $g(\mathscr{W})$, consider the $\trianglelefteq$-optimal modular-map $\mathfrak{g}$. Note that any two modular-states in $\mathfrak{g}(\mathbb{M}_{\equiv})$ are incomparable under $\trianglelefteq$, therefore they either have different ℓ_{Q^L} labels or are non-isomorphic to each other. Since each modular-state has at most

w-many states and since the transition relation in each modular-state is empty, there are at most $(|V|.|Q|)^w \times 2^{n.w^2}$-many isomorphism classes of modular-states. Also $|Q^L| = 2^{\mathcal{O}(|P|.w)}$. It follows from these that $|\mathfrak{g}(\mathbb{M}_{\equiv})| \leq 2^{\mathcal{O}(n.w^2\ log(|V|.|Q|.|P|))}$. Now by Proposition **??** and Theorem 2.3.5 we know that S^g, the domain of the g-retract $g(\mathscr{W})$, satisfies $S^g \subseteq \bigcup_{\mathscr{M} \in \mathfrak{g}(\mathbb{M}_{\equiv})} \mathsf{dom}(\mathscr{M})$. Since each modular-state is of size at most w, the size bound claimed in the theorem follows. □

5.4 Retraction for Modular Witnesses

Now assume for the purposes of this section that the witness $\mathscr{W}$ is modular. The construction of good retractions for $\mathscr{W}$ proceeds in three steps:

1. First we show that if $\mathscr{W}$ admits a (w,m,d)-decomposition, then there exists a retraction g such that $g(\mathscr{W})$ admits a $(w',m,d-1)$-decomposition, for a particular value of w'.
2. Second, using the above result repeatedly, we show that if $\mathscr{W}$ admits a (w,m,d)-decomposition, then there exists a retraction g such that $g(\mathscr{W})$ is w'-simple, for a particular value of w'.
3. Lastly, we combine the above results with Theorem 2.3.10 to show that a $\mathscr{W}$ admits a good retraction.

We begin with the first result mentioned above.

Theorem 5.4.1. *If the witness $\mathscr{W}$ admits a (w,m,d)-decomposition $\equiv_1 \subseteq \ldots \subseteq \equiv_d$, then there exists a retraction g of $\mathscr{W}$ such that,*

1. *the g-retract $g(\mathscr{W})$ is layered-acyclic,*
2. *$g(\mathscr{W})$ admits a $(2^{\mathcal{O}(n.w^2\ log(m.|V|.|Q|.|P|))}, m, d-1)$-decomposition $\equiv_2^g \subseteq \equiv_3^g \subseteq \ldots \subseteq \equiv_d^g$, where $\equiv_c^g \ = \ \equiv_c \cap S^g \times S^g$ for each $2 \leq c \leq d$.*

Proof. Let $\equiv_1 \subseteq \ldots \subseteq \equiv_d$ be a (w,m,d)-decomposition of witness $\mathscr{W}$. For notational convenience, we denote modularization $\equiv_1$ by $\equiv$ and $\equiv_2$ by $\equiv'$.

Let $(\mathbb{M}_{\equiv}, \mathbb{R}_{\equiv}, \mathscr{M}_{\varepsilon})$ and $(\mathbb{M}_{\equiv'}, \mathbb{R}_{\equiv'}, \mathscr{M}'_{\varepsilon})$ denote the respective modular transition systems associated with the modularizations $\equiv$ and $\equiv'$.

By definition of a (w,m,d)-decomposition, the modularization $\equiv$ is w-wide and the modular transition system $(\mathbb{M}_{\equiv}, \mathbb{R}_{\equiv}, \mathscr{M}_{\varepsilon})$ is a tree, and therefore it follows from Theorem 2.3.9 that a $\equiv$-modular progress measure exists. Let $(\mu_{\equiv}, \ell_{Q^L}, \ell_{P^L})$ be a $\equiv$-modular progress measure of the witness $\mathscr{W}$.

Now consider the $\equiv$-modular-preference-order $\trianglelefteq \subseteq \mathbb{M}_{\equiv} \times \mathbb{M}_{\equiv}$ defined as follows:
$\mathscr{M}_1 \trianglelefteq \mathscr{M}_2$, if and only if, there exists an isomorphism h from $\mathscr{M}_1$ to $\mathscr{M}_2$ such that

1. for all $s \in \mathsf{dom}(\mathscr{M}_1)$, we have $\mathscr{N}_{\equiv}(s) = \mathscr{N}_{\equiv}(h(s))$,
2. $\ell_{Q^L}(\mathscr{M}_1) = \ell_{Q^L}(\mathscr{M}_2)$ and $\mu_{\equiv}(\mathscr{M}_1) \succeq_{\ell_{pL}(\mathscr{M}_1)} \mu_{\equiv}(\mathscr{M}_2)$, and
3. $\mathsf{dom}(\mathscr{M}_1) \cup \mathsf{dom}(\mathscr{M}_2) \subseteq \mathsf{dom}(\mathscr{M}')$ for some $\mathscr{M}' \in \mathbb{M}_{\equiv'}$.

Now according to Theorem 2.3.9, for a $\equiv$-modular-preference-order of the form defined above, there exists some $\trianglelefteq$-optimal modular-map. Let $\mathfrak{g}$ be a $\trianglelefteq$-optimal modular-map and let g denote the unbundling of the modular-map $\mathfrak{g}$ that satisfies the condition that $\mathscr{N}_{\equiv}(s) = \mathscr{N}_{\equiv}(g(s))$ for all $s \in S$. It is not difficult to see from the definition of $\trianglelefteq$ and $\mathfrak{g}$ that such an unbundling of $\mathfrak{g}$ exists.

We begin with a simple consequence of the construction of g.

Lemma 5.4.2. *For any $\equiv'$-modular-state $\mathscr{M}' \in \mathbb{M}_{\equiv'}$, we have*

1. *$g(S \cap \mathsf{dom}(\mathscr{M}')) \subseteq g(S) \cap \mathsf{dom}(\mathscr{M}')$, and*
2. *$|S^g \cap \mathsf{dom}(\mathscr{M}')| \leq |g(S) \cap \mathsf{dom}(\mathscr{M}')| = 2^{\mathscr{O}(n.w^2\ log(m.|V|.|Q|.|P|))}$.*

Proof. Consider a $\equiv'$-modular-state $\mathscr{M}' \in \mathbb{M}_{\equiv'}$ and let $\mathbb{M}'_{\equiv} = \{\mathscr{M} \in \mathbb{M}_{\equiv} \mid \mathsf{dom}(\mathscr{M}) \subseteq \mathsf{dom}(\mathscr{M}')\}$. It is an easy consequence of the definition of $\trianglelefteq$ that every $\equiv$-modular state in $\mathbb{M}'_{\equiv}$, is comparable via $\trianglelefteq$, only to other modular states in $\mathbb{M}'_{\equiv}$. Therefore it follows from $\trianglelefteq$-montonicity of $\mathfrak{g}$ that $\mathfrak{g}(\mathbb{M}'_{\equiv}) \subseteq \mathbb{M}'_{\equiv}$.

The first item to be proved here is a simple consequence of the above observation and the fact that g is an unbundling of $\mathfrak{g}$.

For the second item, it suffices to show that $|\mathfrak{g}(\mathbb{M}'_{\equiv})| = 2^{\mathscr{O}(n.w^2\ log(m.|V|.|Q|.|P|))}$, since $|g(S) \cap \mathsf{dom}(\mathscr{M}')|$ is the domain of the modular-states in $\mathfrak{g}(\mathbb{M}'_{\equiv})$, each of which have domain-size at most w. Now since $\mathfrak{g}$ is an $\trianglelefteq$-optimal modular-map, it must be the case that any pair of modular-states in $\mathfrak{g}(\mathbb{M}'_{\equiv})$ are incomparable under $\trianglelefteq$. Note that a pair of modular states are incomparable under $\trianglelefteq$, if either they are non-isomorphic, or the label $\mathscr{N}_{\equiv}$ on states is not preserved, or the label ℓ_{Q^L} is not preserved.

Now note that by definition of a (w,m,d)-decomposition, the number of labels assigned by $\mathscr{N}_{\equiv}$ to states in $\mathscr{M}'$ is bounded by m. It follows that the maximum number of modular-states in $\mathbb{M}'_{\equiv}$ that are either non-isomorphic or do not preserve label $\mathscr{N}_{\equiv}$ on states is bounded by $(m.|V|.|Q|)^w \times 2^{n.w^2}$. Since there are at most $|Q^L|$ different labels for ℓ_{Q^L}, it follows that $|\mathbb{M}'_{\equiv}| = |Q^L| \times (m.|V|.|Q|)^w \times 2^{n.w^2}$. Additionally since $|Q^L| = 2^{\mathscr{O}(|P|.w)}$, we have $|\mathbb{M}'^{k}_{\equiv}| = 2^{\mathscr{O}(n.w^2\ log(m.|V|.|Q|.|P|))}$. □

The fact that the g-retract $g(\mathscr{W})$ is a layered-acyclic witness follows from the fact that $\mathscr{W}$ is layered-acyclic and the first item of Lemma 2.4.2 which says that g maps every state that belongs to a $\equiv_2$-modular-state to another state in the same $\equiv_2$-modular-state.

Next we show that g is a retraction of $\mathscr{W}$. Towards this, consider the Theorem 2.3.5, and observe the following:

- Firstly, note that for any states $s_1, s_2 \in S$ and agent $i \in [n]$, if $s_1 \not\equiv s_2$, then $s_1 \sim_i s_2$, if and only if, $g(s_1) \sim_i g(s_2)$. This is an elementary consequence of the definition of $\mathscr{N}_{\equiv}$ and the fact that $\mathscr{N}_{\equiv}(s) = \mathscr{N}_{\equiv}(g(s))$ for all $s \in S$.

- Secondly observe that for any g-path $s_0a_1s_1a_2...$, the priority sequence $\Omega(\ell_Q(s_0))$, $\Omega(\ell_Q(s_1))$, $\Omega(\ell_Q(s_2))$.. satisfies the parity condition. This follows from the definition of $\trianglelefteq$ and Theorem 2.3.9.

Therefore it follows from item 4 of Theorem 2.3.5 that g is a retraction.

Let $w' = \mathcal{O}(2^{n.w^2 \ log(m.|V|.|Q|.|P|})$. Next we show that $g(\mathscr{W})$ admits the $(w',m,d-1)$-decomposition $\equiv_2^g \subseteq \equiv_3^g \subseteq \ldots \subseteq \equiv_d^g$. To see this, consider the following arguments:

- To see that each $\equiv_c^g$ is a modularization, note that since the function g maps states to other states of the same depth, it follows that for any state $s \in S^g$ with some depth k in the witness $\mathscr{W}$, is also reachable via a length k g-path in $\mathscr{W}$ that begins at the start state of $g(\mathscr{W})$. Since $\equiv_c^g = \equiv_c \cap S^g \times S^g$, it follows that if $s \equiv_c^g$ for some states $s, s' \in S^g$, then the have identical depth in $g(\mathscr{W})$.

  The fact that $\equiv_d^2$ is w'-wide is a simple consequence of Lemma 2.4.2 and the fact that $\equiv_d^g$ is final, follows from $\equiv_d$ being final and the indistinguishability relations $\{\sim_i^g\}_{i\in[n]}$ of the witness $g(\mathscr{W})$ being induced substructures of $\{\sim_i\}_{i\in[n]}$.

- Next we argue that the modular transition system $\mathfrak{M}_{\equiv_2^g}$ associated with the modularization $\equiv_2^g$ of $g(\mathscr{W})$ is a tree. Towards this consider the modular transition systems $\mathfrak{M}_{\equiv_1}, \mathfrak{M}_{\equiv_2}$ associated with modularizations $\equiv_1, \equiv_2$ of the witness $\mathscr{W}$ respectively. Since $\mathscr{W}$ admits a (w,m,d)-decomposition $\equiv_1 \subseteq \ldots \subseteq \equiv_n$, it follows from the definition of a (w,m,d)-decomposition that $\mathfrak{M}_{\equiv}$ is a tree. Moreover, since the modularization $\equiv_2$ is coarser than $\equiv_1$, it also follows that $\mathfrak{M}_{\equiv_2}$ is a tree.

  The fact that $\mathfrak{M}_{\equiv_2^g}$ is a tree, is a consequence of the fact that the modular transition system $\mathfrak{M}_{\equiv_2}$ is a tree and that the retraction g maps every state that belongs to a $\equiv_2$-modular-state to another state in the same $\equiv_2$-modular-state (due to the first item of Lemma 2.4.2).

- Next we show that for every index $2 \leq c < d$, there are at most m states in any equivalence class in $S' \in S^g / \equiv_{c+1}$ with distinct $\equiv_c$-neighbourhoods, that is, $|\{\mathscr{N}_{\equiv_c^g}(s') \mid s' \in$

$S'\}| \leq m$.

Towards this, we claim that for any pair of states $s, s' \in S^g$, if $\mathcal{N}_{\equiv_c}(s) = \mathcal{N}_{\equiv_c}(s')$, then $\mathcal{N}_{\equiv_c^g}(s) = \mathcal{N}_{\equiv_c^g}(s')$. Now if this claim is true, then the our original claim follows, since for any state $s \in S^g$:

$$|\{\mathcal{N}_{\equiv_c^g}(s') \mid s' \in [s]_{\equiv_c^g}\}| \leq |\{\mathcal{N}_{\equiv_c}(s') \mid s' \in [s]_{\equiv_c}\}| \leq m.$$

For a proof of the above claim, consider states $s, s' \in S^g$ that satisfy $\mathcal{N}_{\equiv_c}(s) = \mathcal{N}_{\equiv_c}(s')$. Then by definition of $\mathcal{N}_{\equiv_c}$, we have for every agent $i \in [n]$ that $\mathcal{N}_{\equiv_c}(s)^{\downarrow i} = \mathcal{N}_{\equiv_c}(s')^{\downarrow i}$. Consider the following two cases:

- CASE $\mathcal{N}_{\equiv_c}(s)^{\downarrow i} = \emptyset$: In this case, it follows by definition of $\mathcal{N}_{\equiv_c}$ that $[s]_{\sim_i} \setminus [s]_{\equiv_c} = \emptyset$ holds. Since both $\sim_i^g, \equiv_c^g$ are the respective sub-relations of $\sim_i$, $\equiv_c$ induced by the state-set S^g, it is also the case that $[s]_{\sim_i^g} \setminus [s]_{\equiv_c^g} = \emptyset$, and therefore $\mathcal{N}_{\equiv_c^g}(s)^{\downarrow i} = \emptyset$. Symmetrically, $\mathcal{N}_{\equiv_c^g}(s')^{\downarrow i} = \emptyset$, and therefore, we have $\mathcal{N}_{\equiv_c^g}(s)^{\downarrow i} = \mathcal{N}_{\equiv_c^g}(s')^{\downarrow i}$.
- CASE $\mathcal{N}_{\equiv_c}(s)^{\downarrow i} \neq \emptyset$: Since $\mathcal{N}_{\equiv_c}(s)^{\downarrow i} = \mathcal{N}_{\equiv_c}(s')^{\downarrow i} \neq \emptyset$ holds by assumption, it follows from the definition of $\mathcal{N}_{\equiv_c}$ that $[s]_{\sim_i} = [s']_{\sim_i}$, $[s]_{\sim_i} \setminus [s]_{\equiv_c} \neq \emptyset$ and $[s']_{\sim_i} \setminus [s']_{\equiv_c} \neq \emptyset$. Since both $\sim_i^g, \equiv_c^g$ are the respective sub-relations of $\sim_i, \equiv_c$ induced by the set S^g, it follows easily that $[s]_{\sim_i^g} = [s']_{\sim_i^g}$, $[s]_{\sim_i^g} \setminus [s]_{\equiv_c^g} \neq \emptyset$ and $[s']_{\sim_i^g} \setminus [s']_{\equiv_c^g} \neq \emptyset$. Again from the definition of $\mathcal{N}_{\equiv_c^g}$, it follows that $\mathcal{N}_{\equiv_c^g}(s)^{\downarrow i} = \mathcal{N}_{\equiv_c^g}(s')^{\downarrow i} \neq \emptyset$.

□

Next we combine the observations made in the previous sections and construct good retractions for the class of witnesses that admit a (w, m, d)-decomposition.

Corollary 5.4.3. *If the witness $\mathcal{W}$ admits a (w, m, d)-decomposition $\equiv_1, \ldots, \equiv_d$, then there exists a retraction g such that, the modularization $\equiv_d^g = \equiv_d \cap S^g \times S^g$ of the g-retract $g(\mathcal{W})$ is w^d-wide, where w_d is recursively defined as follows:*

$$w_d := \begin{cases} 2^{\mathcal{O}(n.w_{d-1}^2\ log(m.|V|.|Q|.|P|)}, & if\, d > 1 \\ w, & if\, d = 1. \end{cases}$$

Note that since $\equiv_d$ is a final modularization of $\mathcal{W}$, it follows that $\equiv_d^g$ is a final modularization of $g(\mathcal{W})$ and therefore $g(\mathcal{W})$ is a w_d-simple witness.

Proof. The desired function g is obtained in stages. First we consider the witness $\mathcal{W}$, and apply the Theorem 2.4.1 to obtain a retraction g_1 such that, the g_1-retract $g_1(\mathcal{W})$ admits a $(w_1, m, d-1)$-decomposition where $w_1 = 2^{\mathcal{O}(n.w^2\ log(m.|V|.|Q|.|P|))}$. Now, by Theorem **??** we have that $g_1(\mathcal{W})$ is a witness for the game $(G, \mathcal{Q})$, and by Theorem 2.4.1 the g_1-retract $g_1(\mathcal{W})$ is layered-acyclic and admits a $(w_1, m, d-1)$-decomposition. Therefore we can again use Theorem 2.4.1 to obtain a further retraction g_2 of the witness $g_1(\mathcal{W})$.

Similarly, for any $c \leq d$ we can obtain a retraction g_c of the witness $g_{c-1}(g_{c-2}(..(g_1(\mathcal{W})))$ so that the g_c-retract $g_c(g_{c-1}(g_{c-2}(..(g_1(\mathcal{W}))))$ is a $(w_c, m, d-c)$-decomposition, with $w_c = 2^{\mathcal{O}(w_{c-1}^2 \cdot |V|.|Q|.|\Omega(Q)|^2 + log\ n)}$.

The required retraction g stated in the theorem, is obtained by the composition $g = g_1 \circ g_2 \circ ... \circ g_d$, which by Proposition **??** is guaranteed to be a retraction. The size parameters follows easily from these. □

We conclude by summarizing how to solve the (deterministic) winning strategy problem for game graphs with no (uniform-deterministic) fork-triples.

Theorem 5.4.4.

1. *The winning strategy problem for global μ-automaton winning conditions is solvable for game graphs that contain no fork-triples.*

2. *The deterministic winning strategy problem for global μ-automaton winning conditions is solvable for game graphs that contain no uniform-deterministic fork-triples.*

Proof. Consider an imperfect information game $(G, \mathscr{Q})$ with a game graph G that contain no (uniform-deterministic) fork-triples and global winning conditions given by a μ-automaton $\mathscr{Q}$. By Theorem 2.4.4, the canonical (deterministic) witnesses of such games are modular witnesses, and admit $(1,1,n+1)$-decomposition. Now by Corollary 2.4.1, such witnesses admit retractions that result in a w-simple witnesses, for some computable number w. And using Theorem 2.3.10 we obtain a retraction for this w-simple witness to a bounded-size witness. It therefore follows from solvability Theorem **??** that the (deterministic) winning strategy problem is solvable for global μ-automaton winning conditions with game graphs that contain no (uniform-deterministic) fork-triples. □

5.5 Discussion

We show here that the distributed synthesis problem is solvable for architectures with (weak-)informedness ordering and global μ-automaton specifications. We show this by way of translating them first to games that have game graphs without (uniform-deterministic) fork-triples and have global μ-automaton winning conditions, and then showing that such games are solvable.

Another result which is subsumed by the solvability of games without (uniform-deterministic) fork-triples is from [35], which solves the deterministic winning strategy problem for game graphs that admit the property 'dynamic hierarchic information' and global winning conditions given by priority labellings. It is not difficult to show that the property of having 'dynamic hierarchic information' is equivalent to the property of a game graph having no fork-triples. The solution presented in [35] proceeds by reducing this problem to the distributed synthesis problem for architectures with informedness ordering, and then uses past results on architectures to argue its solvability. In contrast, our result here argues the solvability directly and also extends the solvability of such games to the case of non-deterministic winning strategy problem.

Note that while our final results use only the case when the witnesses admit $(1,1,n+1)$-decomposition, the techiques introduced here allow us to solve classes of games/architectures for which the witnesses admit (w,m,d)-decomposition, for arbitrary $w,m,d \in \mathbb{N}$. Inspite of this technically stronger result, currently we are unaware of any interesting classes of architectures/games for which this result can be utilized. From a purely theoretical point of view, the case of retractions for witnesses that admit (w,m,d)-decompositions, may be seen as partial evidence for the intuition that slight perturbations from the case of (dynamic) hierarchical games (that is, the ones that admits no fork-triples) may still yield a solvable winning strategy problem.

Chapter 6

Synthesis for Local Specifications

In this chapter, we show the following:

1. The *deterministic distributed synthesis problem* is solvable for the class of *weak broadcast architectures* and *local priority labelling specifications*.

2. The *distributed synthesis problem* is solvable for the class of *broadcast architectures* and *local priority labelling specifications*.

Towards showing the above results, we divide our analyses into the following steps:

1. First, we show that for any game $(G, \{\gamma_i\}_{i \in [n]})$, with the game graph G corresponding to (weak) broadcast architectures, and winning conditions given by local priority labelling conditions, the canonical witnesses of $(G, \{\gamma_i\}_{i \in [n]})$ satisfy the 'broadcast property'.

2. Second, we define a notion of 'factorization' of a witness, which is comprised of two parts. The first part of a factorization is comprised of the set of 'factors' of the witness, and the second is a 'coupling relation' that records how the various factors interact.

We show that the canonical witnesses of $(G, \{\gamma_i\}_{i\in[n]})$ of the form given above admit a factorization into factors that are similar to modular witnesses. Additionally, we show that the coupling relation satisfies a property called the 'grid property'.

3. Third, we define a notion of retraction of a factor, and show that the retractions of factors may be used to construct retractions of the witness, if they satisfy some 'compatibility' conditions.

   Moreover, we construct a retraction for canonical witnesses of $(G, \{\gamma_i\}_{i\in[n]})$, using the retractions of factors that satisfy the compatibility relation. It is crucial for this construction that the factors of the witness are similar to modular witnesses, and that the coupling relation satisfies the grid property.

4. Lastly we show that there exists a computable function f such that, any witness $\mathscr{W}$ of any game $(G, \{\gamma_i\}_{i\in[n]})$ of the form given above admits a retraction g with $g(\mathscr{W})$ of size at most $f(|G|, |\{\gamma_i\}_{i\in[n]}|)$.

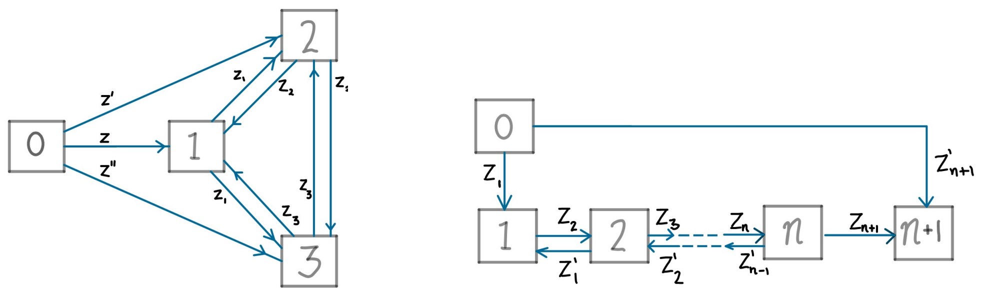
By the retraction criterion mentioned in Chapter 4, the solvability of the distributed synthesis problems above follows.

6.1 Games for Broadcast Architectures

We begin by fixing some notation. Let $P = \{1, 2, .., |P|\}$ be a set of priorities. Let $(G, \{\gamma_i\}_{i\in[n]})$ be an imperfect information game, where $G = (V, E, v_\varepsilon)$ is a game graph and $\{\gamma_i\}_{i\in[n]}$ is a local priority labelling winning condition with each γ_i as a function that maps states in V_i to priorities in P. Let $\mathfrak{N} = \{N_1, N_2, .., N_d\}$ be a partitioning of agents in $[n]$.

Now consider an architecture $\mathsf{Ar} = ([n]^+, \alpha)$ on the set of variables Z. Recall that for any agent $i \in [n]$, In_i and Out_i denote the set of input and output variables of agent i, respectively. For any subset $N \subseteq [n]$, let $\mathsf{In}_N = \bigcup_{i\in N} \mathsf{In}_i$ and $\mathsf{Out}_N = \bigcup_{i\in N} \mathsf{Out}_i$.

An architecture Ar is said to be a *(weak)* $\mathfrak{N}$*-broadcast architecture*, if



(a) A simple broadcast architecture (b) A Pipeline with feedback

Figure 6.1: Broadcast Architectures: Examples

1. each agent-set $N_c \in \mathfrak{N}$ admits a (weak-)informedness ordering[1].

2. for any index $c \in [d]$ and agent $i \in [n] \setminus N_c$, we have $\mathsf{Out}_{N_c} \cap \mathsf{In}_{[n]\setminus N_c} \subseteq \mathsf{In}_i$.

3. for any index $c \in [d]$, we have $\mathsf{Out}_{N_c} \cap \mathsf{In}_{[n]\setminus N_c} \subseteq \mathsf{Out}_{i_c}$, where i_c denotes the least informed agent in N_c, under the (weak-)informedness ordering.

We use the term *(weak) broadcast architecture*, to refer to a *(weak) $\mathfrak{N}$-broadcast architecture* for some partitioning $\mathfrak{N}$.

Next we give an intuitive description of the above conditions. The first condition is self-explanatory. The second condition says whenever information is transmitted from an agent in a partition N_c to an agent outside the partition, then it is also transmitted to every other agent in the system. The third condition says that in a partition N_c, only the least informed agent i_c is allowed to transmit outside of N_c.

An interesting special case of the above class of architectures is that of 'simple broadcast architecture' where every agent transmits equally to all other agents. Formally, we say that an architecture Ar is a *simple broadcast architecture* if $\mathsf{Out}_i \cap \mathsf{In}_{[n]\setminus\{i\}} \subseteq \mathsf{In}_j$ for any agents $i \in [n]$ and $j \in [n] \setminus \{i\}$. It is easy to verify that this is a special case of (weak) $\mathfrak{N}$-broadcast architecture where $\mathfrak{N} = \{\{1\},\{2\},..,\{n\}\}$.

[1]Refer to Section 2.1 to recall this definition.

Comparison with Earlier Results As mentioned in Chapter 2, the previous state-of-the-art result for solvability in local specifications, is for flanked pipelines with feedback architectures and local ω-automaton specifications. In order to define these architecture classes, we define some terminology. We qualify an architecture Ar as

- a *pipeline*, if there exist an enumeration $i_1, i_2, ..i_n$ of agent in n such that $\mathsf{In}_i = \mathsf{Out}_{i-1}$ for all $1 \leq i \leq n$. We call this enumeration the *pipeline order*.
- a *pipeline with feedback*, if the architecture is obtained by considering a pipeline architecture and adding edges from agents that appear later in the the pipeline order to the ones that appear earlier in the order.

Note that pipeline architectures and pipeline with feedback architectures are instances of architectures that admit a weak-informedness ordering, and this can be verified easily.

Flanked pipelines are a special case of weak broadcast architecture that admits a partitioning into two sub-architectures, where the first sub-architecture is a pipeline, the second sub-architecture is a single agent, and the least informed agent in the first sub-architecture can communicate with the second sub-architecture. Flanked pipelines with feedback are similar to the above, with the modification that the first sub-architecture now consists of a pipeline with feedback.

We note here that, in this chapter we show the solvability for these architectures for priority labelling specifications, and not the more general ω-automaton specifications. We will discuss the reasons for this at the end.

Let $\mathscr{W} = (H, M, \{\approx_i\}_{i \in [n]}, \{\ell_V\}, v_\varepsilon)$ be a (deterministic) canonical witness of $(G, \{\gamma_i\}_{i \in [n]})$, associated with a winning (deterministic) joint strategy σ.[2]

Next we state a few properties of the canonical (deterministic) witness $\mathscr{W}$, due to being

[2]Note the change in convention; earlier the canonical witness associated with σ was denoted by $(H^\sigma, M^\sigma, \{\approx_i^\sigma\}_{i \in [n]}, \{\ell_V\}, v_\varepsilon)$ and the game tree specified by G was denoted by $(H, M, \{\approx_i\}_{i \in [n]}, v_\varepsilon)$.

obtained from a (weak) broadcast architecture. The latter property in the theorem given below is in some sense a true characteristic of a broadcast systems, and we informally refer to this as the *broadcast property*.

Theorem 6.1.1. *Let Ar be a (weak) $\mathfrak{N}$-broadcast architecture and let G be a game graph corresponding to Ar.*

1. *If $N_c \in \mathfrak{N}$ with $i_1, i_2, ..i_k$ denoting the (weak-)informedness ordering on the agents in N_c, then for any history $\tau \in H$, we have $[\tau]_{\approx_{i_1}} \subseteq [\tau]_{\approx_{i_2}} \subseteq .. \subseteq [\tau]_{\approx_{i_k}}$.*

2. *For any histories $\tau_1 a_1 v_1, \tau_2 a_2 v_2 \in H$, if $\tau_1 a_1 v_1 \approx_i \tau_2 a_2 v_2$ for some $i \in [n]$, then $a_1^{\downarrow i_c \downarrow z} = a_2^{\downarrow i_c \downarrow z}$ for every index $c \in [d]$ and variable $z \in Out_{N_c} \cap In_{[n] \setminus N_c}$.*

Proof. 1. To show the first claim, it suffices to show that for any histories $\tau_1, \tau_2 \in H$ and agent i_j with $j < k$, if $\tau_1 \approx_{i_j} \tau_2$, then $\tau_1 \approx_{i_{j+1}} \tau_2$.

We argue the above claim by induction on the sum of length of the histories τ_1, τ_2. For the base case, both these histories correspond to the trivial history v_ε, and the claim follows trivially. For the induction case, consider histories $\tau_1 a_1 v_1, \tau_2 a_2 v_2 \in H$ and agent i_j with $j < k$, such that $\tau_1 a_1 v_1 \approx_{i_j} \tau_2 a_2 v_2$ holds. We need to show that $\tau_1 a_1 v_1 \approx_{i_{j+1}} \tau_2 a_2 v_2$.

Now by the perfect recall property satisfied by the canonical witness $\mathscr{W}$, it follows from $\tau_1 a_1 v_1 \approx_{i_j} \tau_2 a_2 v_2$ that $\tau_1 \approx_{i_j} \tau_2$. And additionally, by induction hypothesis on the shorter length histories τ_1, τ_2, we have that $\tau_1 \approx_{i_{j+1}} \tau_2$.

Now assume for a contradiction that $\tau_1 a_1 v_1 \not\approx_{i_{j+1}} \tau_2 a_2 v_2$, then since $\tau_1 \approx_{i_{j+1}} \tau_2$ holds, it follows from the definition of $\approx_{i_{j+1}}$ that either $a_1^{\downarrow i_{j+1}} \neq a_2^{\downarrow i_{j+1}}$ or $v_1^{\downarrow i_{j+1}} \neq v_2^{\downarrow i_{j+1}}$. The argument now splits into the following cases:

- $i_1, i_2, ..i_k$ is a weak-informedness ordering and $\mathscr{W}$ is canonical *deterministic* witness: Since $\mathscr{W}$ is uniform and deterministic, and since $\tau_1 \approx_{i_{j+1}} \tau_2$ holds, it follows that the actions of agent i_{j+1} in the joint actions a and a' are identical, and so $a_1^{\downarrow i_{j+1}} = a_2^{\downarrow i_{j+1}}$. Therefore it must be the case that $v_1^{\downarrow i_{j+1}} \neq v_2^{\downarrow i_{j+1}}$ holds, or equivalently, that there exists a variable $z \in \mathsf{In}_{i_{j+1}}$ such that $v_1^{\downarrow i_{j+1}\downarrow z} \neq v_2^{\downarrow i_{j+1}\downarrow z}$.

Now by definition of weak-informedness $\mathsf{In}_{i_j} \cup \mathsf{Out}_{i_j} \subseteq \mathsf{In}_{i_{j+1}}$, that is, the variable z read by agent i_{j+1}, is read or written on by agent i_j. Therefore it follows from the construction of the game graph G, that either $a_1^{\downarrow i_j\downarrow z} \neq a_2^{\downarrow i_j\downarrow z}$ or $v_1^{\downarrow i_j\downarrow z} \neq v_2^{\downarrow i_j\downarrow z}$.

- $i_1, i_2, ..i_k$ is an informedness ordering and the witness $\mathscr{W}$ is canonical witness: Since either $a_1^{\downarrow i_{j+1}} \neq a_2^{\downarrow i_{j+1}}$ or $v_1^{\downarrow i_{j+1}} \neq v_2^{\downarrow i_{j+1}}$ hold, it must be the case that either there is a variable $z \in \mathsf{Out}_{i_{j+1}}$ such that $v_1^{\downarrow i_{j+1}\downarrow z} \neq v_2^{\downarrow i_{j+1}\downarrow z}$ or there is a variable $z \in \mathsf{In}_{i_{j+1}}$ such that $v_1^{\downarrow i_{j+1}\downarrow z} \neq v_2^{\downarrow i_{j+1}\downarrow z}$.

  Now by definition of informedness $\mathsf{In}_{i_j} \cup \mathsf{Out}_{i_j} \supseteq \mathsf{In}_{i_{j+1}} \cup \mathsf{Out}_{i_{j+1}}$, that is, the variable z read or written on by agent i_{j+1}, is read or written on by agent i_j. It follows from the construction of the game graph G, that either $a_1^{\downarrow i_j\downarrow z} \neq a_2^{\downarrow i_j\downarrow z}$ or $v_1^{\downarrow i_j\downarrow z} \neq v_2^{\downarrow i_j\downarrow z}$.

Therefore in either of the above cases we have by definition of the game graph G that $a_1^{\downarrow i_j} \neq a_2^{\downarrow i_j}$ or $v_1^{\downarrow i_j} \neq v_2^{\downarrow i_j}$. It follows from the definition of the indistinguishability relation $\approx_i$ that $\tau_1 a_1 v_1 \not\approx_{i_j} \tau_2 a_2 v_2$, a contradiction to one of our assumptions. This completes the proof of the first item of the theorem.

2. Let $\tau_1 a_1 v_1 \approx_i \tau_2 a_2 v_2$ for an agent $i \in [n]$, and assume for a contradiction that $a_1^{\downarrow i_c\downarrow z} \neq a_2^{\downarrow i_c\downarrow z}$ for some index $c \in [d]$ and variable $z \in \mathsf{Out}_{N_c} \cap \mathsf{In}_{[n]\setminus N_c}$. We consider the following mutually exclusive and exhaustive cases:

   - If $i \in N_c$, then due to i_c being the least informed agent in N_c and due to Proposition 3.1.1, it follows from $\tau_1 a_1 v_1 \approx_i \tau_2 a_2 v_2$ that $\tau_1 a_1 v_1 \approx_{i_c} \tau_2 a_2 v_2$. Now due to the definition of $\approx_{i_c}$, it must be the case that $a_1^{\downarrow i_c} = a_2^{\downarrow i_c}$, which contradicts our assumption that $a_1^{\downarrow i_c\downarrow z} \neq a_2^{\downarrow i_c\downarrow z}$.
   - If $i \notin N_c$, then let $N_{c'}$ be a partition with $i \in N_{c'}$. Firstly note that since $z \in \mathsf{Out}_{N_c} \cap \mathsf{In}_{[n]\setminus N_c}$, it is the case that z is a variable written on by i_c and read by

some agent in $[n] \setminus N_c$. It follows from the definition of (weak-) broadcast architectures that z is read by every agent in $[n] \setminus N_c$, and in particular, that z is read by agent i.

Now since agent i reads different values for z after the actions a_1, a_2, it follows from the construction of the game graph G that $v_1^{\downarrow i \downarrow z} \neq v_2^{\downarrow i \downarrow z}$, and therefore, $v_1^{\downarrow i} \neq v_2^{\downarrow i}$. It now follows from the definition of $\approx_i$ that $\tau_1 a_1 v_1 \not\approx_i \tau_2 a_2 v_2$, a contradiction to our assumption.

$\square$

6.2 Factorizing Witnesses

In this section we define the notion of factorization of a witness, and show that canonical (deterministic) witnesses of the games obtained from (weak) broadcast architectures admit a factorization into 'factors' that are similar to modular witnesses. We also show that these factors are 'coupled' in a way that is favourable to our needs.

For the remainder of this chapter, let us fix a witness $\mathscr{S} = (S, R, \{\sim_i\}_{i \in [n]}, \{\ell_V\}, s_\varepsilon)$ of the game $(G, \{\gamma_i\}_{i \in [n]})$. For any subset $N \subseteq [n]$ and state $s \in S$, let $[s]^N$ denote the tuple $([s]_{\sim_i})_{i \in N}$ of equivalence classes[3], and let $V^N = \prod\limits_{i \in N} V_i$ and $A^N = \prod\limits_{i \in N} A_i$.

N-factor The *N-factor* of the witness $\mathscr{S}$ is given by the $(\{V^N\}, \{A_i\}_{i \in N})$-MaLTS $\mathscr{S}^N = (S^N, R^N, \{\sim_i^N\}_{i \in N}, \{\ell_{V^N}\}, [s_\varepsilon]^N)$ where:

1. $S^N := \{(([s]^N \mid s \in S\}$.
2. The transition relation $R^N \subseteq S^N \times A^N \times S^N$ is such that, $([s_1]^N, a^N, [s_2]^N) \in R_N$, if and only if, there exists a transition $(s_1', a, s_2') \in R$ such that $s_1' \in [s_1]^N$, $s_2' \in [s_2]^N$ and

[3]Analogous to this, for any history $\tau \in H$ in the canonical witness $\mathscr{W}$, we denote a tuple $([\tau]_{\sim_i})_{i \in N}$ by $[\tau]^N$.

$a^{\downarrow i} = a^{N\downarrow i}$ holds for all agents $i \in N$.

3. For any agent $i \in N$, the equivalence relation $\sim_i^N \subseteq S^N \times S^N$ is such that $[s_1]^N \sim_i^N [s_2]^N$, if and only if, $[s_1]^{N\downarrow i} = [s_2]^{N\downarrow i}$.

4. The labellings $\ell_{V^N} : S^N \to V^N$ is defined as follows: for any $s \in S$, we have $\ell_{V^N}([s]^N)^{\downarrow i} := \ell_V(s)^{\downarrow i}$ for agents $i \in N$.

   Here ℓ_{V^N} is well-defined due to condition $\mathbf{S}'$.3 satisfies by the witness $\mathscr{W}$.

Factorization and Coupling Relation An $\mathfrak{N}$-*factorization* of the witness $\mathscr{S}$ is a tuple $(\{\mathscr{S}^{N_c}\}_{c\in[d]}, \mathscr{C})$, where $\mathscr{S}^{N_c}$ is an N_c-factor of $\mathscr{S}$, for each $c \in [d]$, and $\mathscr{C} := \{([s]^{N_c})_{c\in[d]} \mid s \in S\}$. We call $\mathscr{C}$ the *coupling relation* of the $\mathfrak{N}$-factorization.

Intuitively, the presence of a tuple $([s_1]^{N_1}, ([s_2]^{N_2}, ..., ([s_d]^{N_d})$ in the coupling relation indicates that there is some state $s \in S$ such that for each $c \in [d]$, we have $[s]^{N_c} = [s_c]^{N_c}$.

We begin by stating an elementary property of the witness $\mathscr{S}$ which is an immediate consequence of the Unique-Type condition satisfied by the witness: for all states $s_1, s_2 \in S$, there exists an agent $i \in [n]$ such that $s_1 \not\sim_i s_2$.

Proposition 6.2.1. *For any sequence of equivalence classes $[s_1]^1, [s_2]^2, .., [s_n]^n$, one for each agent in $[n]$, $|\bigcap_{i\in[n]} [s_i]^i| \leq 1$.*

It is a simple consequence of the definition of N-factors and Proposition 3.2.1 that, every witness $\mathscr{S}$ is isomorphic to its $[n]$-factor $\mathscr{S}^{[n]}$. An N-factor of a witness $\mathscr{S}$ may now be seen as a projection of the structure $\mathscr{S}^{[n]}$, to the components in N.

While an N-factor of a witness does not retain all the properties of a witness, it preserves some key properties. We combine these properties in the next definition.

We say that a $(\{V^N\}, \{A_i\}_{i\in N})$-MaLTS $\mathscr{S}^N = (S^N, R^N, \{\sim_i^N\}_{i\in N}, \{\ell_{V^N}\}, [s_\varepsilon]^N)$ a *witness-like*, if it satisfies the following

S$'$.3. $\forall [s_1]^N, [s_2]^N \in S^N, \forall i \in N : [s_1]^N \sim_i^N [s_2]^N$ implies $\ell_{V^N}([s_1]^N)^{\downarrow i} = \ell_{V^N}([s_2]^N)^{\downarrow i}$.

S$'$.4. $\forall [s_1]^N, [s_2]^N \in S^N, \exists i \in N : [s_1]^N \not\sim_i^N [s_2]^N$.

V.2 $\forall i \in N, \forall ([s_1]^N, a^N, [s_2]^N), ([s_1']^N, a'^N, [s_2']^N) \in R^N :$ if $[s_1]^N \sim_i^N [\tau_1']^N$, then, $[s_2]^N \sim_i^N [s_2']^N$, if and only if, $\ell_{V^N}([s_2]^N)^{\downarrow i} = \ell_{V^N}([s_2']^N)^{\downarrow i}$ and $a^{N\downarrow i} = a'^{N\downarrow i}$.

Q$'$.1. for every path $[s_0]^N a_1^N [s_1]^N a_2^N ..$ and agent $i \in N$, the sequence $\gamma_i(\ell_{V^N}([s_0]^N)^{\downarrow i}), \gamma_i(\ell_{V^N}([s_1]^N)^{\downarrow i}), ..$ satisfies the parity condition.

The proof of theorem below follows straightforwardly from the definition of a witness and its N-factor.

Theorem 6.2.2. *For any $N \in [n]$, the N-factor of the witness $\mathscr{S}$ is witness-like.*

Next we show that the canonical witnesses for games obtained from broadcast architectures admit a factorization into parts that are modular MaLTS's.

Theorem 6.2.3. *Let Ar be a (weak) $\mathfrak{N}$-broadcast architecture and let G be a game graph corresponding to Ar. Then for any $c \in [d]$, the N_c-factor $\mathscr{W}^{N_c}$ of the (deterministic) canonical witness $\mathscr{W}$ admits a sharp $(1, 1, |N_c| + 1)$-decomposition*[4].

Proof. Consider the N_c-factor $\mathscr{W}^{N_c}$ of the witness $\mathscr{W}$, and let $i_1, i_2, .., i_k$ denote (weak-)informedness order on agents in N_c. Let $\mathscr{H}^{N_c} = \bigcup_{i \in N_c} H^{N_c} / \approx_i^{N_c}$.

The proof of this follows along lines similar to those of Theorem 2.4.4. We begin by showing that $(\mathscr{H}^{N_c}, \subseteq)$ is a tree-order, that is, $\emptyset \notin \mathscr{H}^{N_c}$ and for any distinct elements $t_1, t_2 \in \mathscr{H}^{N_c}$, either $t_1 \subset t_2$ or $t_2 \subset t_1$ or $t_1 \cap t_2 = \emptyset$. The fact that $\emptyset \notin \mathscr{H}^{N_c}$ is easily seen to be true. To see the remaining claim, note that by Proposition 3.1.1, $[\tau]_{\approx_{i_1}} \subseteq [\tau]_{\approx_{i_2}} \subseteq .. \subseteq [\tau]_{\approx_{i_k}}$, for any history τ in the canonical (deterministic) witness $\mathscr{W}$. It is an easy consequence of this fact and the definitions of $\approx_{i_1}^{N_c}, \approx_{i_2}^{N_c}, .., \approx_{i_k}^{N_c}$ that, $[[\tau]^{N_c}]_{\approx_{i_1}^{N_c}} \subseteq [[\tau]^{N_c}]_{\approx_{i_2}^{N_c}} \subseteq .. \subseteq [[\tau]^{N_c}]_{\approx_{i_k}^{N_c}}$, for any element $[\tau]^{N_c} \in H^{N_c}$, in the factor $\mathscr{W}^{N_c}$. The claim follows from this observation.

[4]Refer to Section 2.2 for the definition of *sharp modularization*.

The desired sharp $(1, 1, |N_c| + 1)$-decomposition of the N_c-factor $\mathscr{W}^{N_c}$ can now be constructed using arguments similar to those in Theorem 2.4.4. □

Next we show a characteristic property of the coupling relation of the $\mathfrak{N}$-factorization of the witness $\mathscr{W}$, informally referred to as the *grid property*.

Theorem 6.2.4. *Let Ar be a (weak) $\mathfrak{N}$-broadcast architecture and let G be a game graph corresponding to Ar. Let $\equiv$ be the sharp final modularization of the canonical witness $\mathscr{W}$, and let $\mathfrak{M}_\equiv = (\mathbb{M}_\equiv, \mathbb{R}_\equiv, \mathscr{M}_\varepsilon)$ be the modular transition system associated with $\equiv$. Let $\mathscr{C}$ be the coupling relation of the $\mathfrak{N}$-factorization of $\mathscr{W}$.*

Then, $\mathscr{C} = \bigcup\limits_{\mathscr{M} \in \mathbb{M}_\equiv} \mathscr{C}_\mathscr{M}$, where $\mathscr{C}_\mathscr{M} = \prod\limits_{c \in [d]} \bigcup\limits_{\tau \in dom(\mathscr{M})} [\tau]^{N_c}$ for each $\mathscr{M} \in \mathbb{M}_\equiv$.

Proof. The forward direction $\mathscr{C} \subseteq \bigcup\limits_{\mathscr{M} \in \mathbb{M}_\equiv} \mathscr{C}_\mathscr{M}$ holds, since $\mathscr{C} := \{([\tau]^{N_c})_{c\in[d]} \mid \tau \in H\}$ holds by definition of the coupling relation $\mathscr{C}$.

For the reverse direction, that is, to show $\mathscr{C} \supseteq \bigcup\limits_{\mathscr{M} \in \mathbb{M}_\equiv} \mathscr{C}_\mathscr{M}$, it suffices to show that $\mathscr{C}_\mathscr{M} \subseteq \mathscr{C}$ holds for any modular-state $\mathscr{M}$. Towards this, consider an arbitrary modular-state $\mathscr{M}$. We prove the claim $\mathscr{C}_\mathscr{M} \subseteq \mathscr{C}$ by induction on the length of the unique modular-path that begins at $\mathscr{M}_\varepsilon$ and ends at $\mathscr{M}$. Note that since $\mathscr{W}$ is a canonical witness, it must be the case that $\mathfrak{M}_\equiv$ is a tree, and such unique modular-path exists between $\mathscr{M}_\varepsilon$ and $\mathscr{M}$.

For the base case, consider the modular-state $\mathscr{M}_\varepsilon$ which contains exactly one history v_ε. The fact that $([v_\varepsilon]^{N_c})_{c\in[d]} \in \mathscr{C}$ follows from the definition of $\mathscr{C}$, and therefore $\mathscr{C}_{\mathscr{M}_\varepsilon} \subseteq \mathscr{C}$. Towards the inductive case, assume that the claim holds true for a modular-state $\mathscr{M}$, that is, $\mathscr{C}_\mathscr{M} \subseteq \mathscr{C}$, and consider a modular-state $\mathscr{M}' \in \mathbb{M}_\equiv$ that has some incoming transition $(\mathscr{M}, R', \mathscr{M}')$ from $\mathscr{M}$. We need to show that $\mathscr{C}_{\mathscr{M}'} \subseteq \mathscr{C}$.

We begin by defining some notation.

- For any history $\tau \in H$, let $([\tau]^{N_c})_{[d]}$ denote the tuple $([\tau]^{N_c})_{c\in[d]}$.
- For any $D \subseteq [d]$, let $\mathsf{swap}_D(([\tau_1]^{N_c})_{[d]}, ([\tau_2]^{N_c})_{[d]})$ denote the operation that takes the

two tuples $([\tau_1]^{N_c})_{[d]}$ and $([\tau_2]^{N_c})_{[d]}$ as input, and returns the tuple $(X_c)_{c\in[d]}$ with $X_c = [\tau_2]^{N_c}$ for each $c \in D$ and $X_c = [\tau_1]^{N_c}$ for each $c \in [d] \setminus D$.

Returning to our main claim, we argue that following two claims imply $\mathscr{C}_{\mathscr{M}'} \subseteq \mathscr{C}$:

1. For any two histories $\tau_1', \tau_2' \in \mathsf{dom}(\mathscr{M}')$, the set of tuples $\prod_{c\in[d]} \{[\tau_1']^{N_c}, [\tau_2']^{N_c}\} \subseteq \mathscr{C}$.
2. For any two histories $\tau_1', \tau_2' \in \mathsf{dom}(\mathscr{M}')$ and for every $c \in [d]$, the tuple $\mathsf{swap}_{\{c\}}(([\tau_1']^{N_c})_{[d]}, ([\tau_2']^{N_c})_{[d]})$ belongs to $\mathscr{C}$.

Observe that if the first claim were true, then our desired result follows. To see this, observe that if $\prod_{c\in[d]} \{[\tau_1']^{N_c}, [\tau_2']^{N_c}\} \subseteq \mathscr{C}$ for any two histories $\tau_1', \tau_2' \in \mathsf{dom}(\mathscr{M}')$, then it follows that $\prod_{c\in[d]} \{[\tau']^{N_c} \mid \tau' \in \mathsf{dom}(\mathscr{M}')\} \subseteq \mathscr{C}$, or equivalently, that $\mathscr{C}_{\mathscr{M}} \subseteq \mathscr{C}$.

Next we argue that the second claim implies the first claim. Towards this assume that the second claim is true and observe the following expressions.

- firstly observe that $\prod_{c\in[d]} \{[\tau_1']^{N_c}, [\tau_2']^{N_c}\} = \bigcup_{D\subseteq[d]} \mathsf{swap}_D(([\tau_1']^{N_c})_{[d]}, ([\tau_2']^{N_c})_{[d]})$, and
- secondly, for any $D \subseteq [d]$ with $D = \{c_1, c_2, .., c_k\}$, we have the following:

$$\mathsf{swap}_D\big(([\tau_1']^{N_c})_{[d]}, ([\tau_2']^{N_c})_{[d]}\big)$$
$$=$$
$$\mathsf{swap}_{\{c_1\}}\Big(...\mathsf{swap}_{\{c_{k-1}\}}\Big(\mathsf{swap}_{\{c_k\}}\Big(([\tau_1']^{N_c})_{[d]}, ([\tau_2']^{N_c})_{[d]}\Big), ([\tau_2']^{N_c})_{[d]}\Big), ..., ([\tau_2']^{N_c})_{[d]}\Big).$$

The correctness of both the above expressions follows straightforwardly from the definition of swap_D. From our assumption about the correctness of the second claim, one can argue inductively, starting at the innermost parentheses in the second expression above, that each operation $\mathsf{swap}_{\{c_j\}}$ returns a tuple in the coupling relation $\mathscr{C}$. It follows immediately that $\mathsf{swap}_D\big(([\tau_1']^{N_c})_{[d]}, ([\tau_2']^{N_c})_{[d]}\big)$ is a tuple in the coupling relation $\mathscr{C}$. Combining this observation with the first expression, we obtain that the tuples in $\prod_{c\in[d]} \{[\tau_1']^{N_c}, [\tau_2']^{N_c}\}$ belong to $\mathscr{C}$, which is the first claim.

We can conclude form the above discussion that to show our main claim, it suffices to show that for any two histories $\tau_1', \tau_2' \in \mathsf{dom}(\mathscr{M}')$ and an arbitrary agent, say 1, the tuple $\mathsf{swap}_{\{1\}}(([\tau_1']^{N_c})_{[d]}, ([\tau_2']^{N_c})_{[d]})$, or equivalently, the tuple $([\tau_2']^{N_1}, [\tau_1']^{N_2}, .., [\tau_1']^{N_d})$ belongs to the coupling relation $\mathscr{C}$. By definition of the coupling relation $\mathscr{C}$, this is equivalent to showing that there exists a history $\tau' \in \mathsf{dom}(\mathscr{M}')$, such that $([\tau']^{N_1}, [\tau']^{N_2}, .., [\tau']^{N_d}) = ([\tau_2']^{N_1}, [\tau_1']^{N_2}, .., [\tau_1']^{N_d})$. In the remaining part of this proof, we show that such a history τ' exists.

Towards this, observe that since the modular-state $\mathscr{M}'$ has some incoming transition $(\mathscr{M}, R', \mathscr{M}')$ from $\mathscr{M}$, it must be the case that there exist histories $\tau_1, \tau_2 \in \mathsf{dom}(\mathscr{M})$ such that $\tau_1' = \tau_1 a_1 v_1$ and $\tau_2' = \tau_2 a_2 v_2$ for some $a_1, a_2 \in A$ and $v_1, v_2 \in V$. From the inductive assumption $\mathscr{C}_{\mathscr{M}} \subseteq \mathscr{C}$, it follows that the tuples $([\tau_1]^{N_1}, [\tau_1]^{N_2}, .., [\tau_1]^{N_d})$, $([\tau_2]^{N_1}, [\tau_2]^{N_2}, .., [\tau_2]^{N_d})$ and $([\tau_2]^{N_1}, [\tau_1]^{N_2}, .., [\tau_1]^{N_d})$ belong to the coupling relation $\mathscr{C}$. It follows from the definition of the coupling relation $\mathscr{C}$, that there exists a history τ such that $([\tau]^{N_1}, [\tau]^{N_2}, .., [\tau]^{N_d}) = ([\tau_2]^{N_1}, [\tau_1]^{N_2}, .., [\tau_1]^{N_d})$, and moreover, from Proposition 3.2.1 such a history τ is unique.

We claim next that the desired history τ' is the history $\tau a v$ that satisfies the following:

- $a^{\downarrow i} = a_2^{\downarrow i}$ and $v^{\downarrow i} = v_2^{\downarrow i}$ for every agent $i \in N_1$, and
- $a^{\downarrow i} = a_1^{\downarrow i}$ and $v^{\downarrow i} = v_1^{\downarrow i}$ for every agent $i \in [n] \setminus N_1$.

It is easy to see that if such a history τ' belongs to the witness $\mathscr{W}$, then from the definition of τ' and $\approx_i$ we have the following: firstly, $\tau a v \approx_i \tau a_2 v_2$ for all $i \in N_1$; and secondly $\tau a v \approx_i \tau a_1 v_1$ for every index $c \in [d] \setminus \{1\}$ and agent $i \in N_c$. It follows that $([\tau']^{N_1}, [\tau']^{N_2}, .., [\tau']^{N_d}) = ([\tau_2']^{N_1}, [\tau_1']^{N_2}, .., [\tau_1']^{N_d})$. Moreover, since $\tau a_2 v_2$ is a state in $\mathsf{dom}(\mathscr{M}')$, and since $\tau a v \approx_i \tau a_2 v_2$ for some $i \in N_1$, it follows from the definition of a final modularization that $\tau a v \equiv \tau a_2 v_2$, and therefore $\tau a v \in \mathsf{dom}(\mathscr{M}')$.

And so it suffices to show that the history $\tau' = \tau a v$ belongs to the witness $\mathscr{W}$. We argue this in two parts. Before we begin, note that by construction of the canonical witness $\mathscr{W}$, the MaTS underlying $\mathscr{W}$ is identical to the MaTS underlying the extended strategy tree of the joint strategy σ, and therefore by Proposition **??** the MaLTS $\mathscr{W}$ is uniform and strategic. We will use this fact in the arguments to follows.

- Firstly we show that there exists an outgoing a-transition at τ such that $a^{\downarrow i} = a_2^{\downarrow i}$ for every agent $i \in N_1$, and $a^{\downarrow i} = a_1^{\downarrow i}$ for every agent $i \in [n] \setminus N_1$.

  To see this, observe that since $[\tau]^{N_1} = [\tau_2]^{N_1}$ and $[\tau]^{N_c} = [\tau_1]^{N_c}$ for all $c \in [d] \setminus \{1\}$, it follows that $\tau \approx_i \tau_2$ for all agents $i \in N_1$, and that $\tau' \approx_i \tau_2'$ for all agents $i \in [n] \setminus N_1$. Now since the histories τ_1, τ_2 have prolongations $\tau_1 a_1 v_1, \tau_2 a_2 v_2$ respectively, it follows from the witness $\mathscr{W}$ being a uniform MaLTS, that there exists some joint action $a' \in A$ and an outgoing a'-transition at τ such that $a'^{\downarrow i} = a_2^{\downarrow i}$ for every agent $i \in N_1$ and $a'^{\downarrow i} = a_1^{\downarrow i}$ for every agent $i \in [n] \setminus N_1$. The joint action a is exactly the joint action a' whose existence is argued above.

- Secondly, we show that there exists a transition $(\tau, a, \tau a v)$ in the witness $\mathscr{W}$, with the joint action $a \in A$ satisfying the above properties, and v such that $v^{\downarrow i} = v_2^{\downarrow i}$ for every agent $i \in N_1$ and $v^{\downarrow i} = v_1^{\downarrow i}$ for every agent $i \in [n] \setminus N_1$.

  Towards this, firstly note that since there exists an outgoing a-transition at τ, it follows from the witness $\mathscr{W}$ being a strategic MaLTS that, the set of histories $H' = \{\tau a v' \mid v' \in \mathsf{end}(\tau) E_a\}$ also belongs to the witness $\mathscr{W}$. Note that the set of states $\mathsf{end}(\tau) E_a$ is determined only by the action a, and the following is a simple consequence of the construction of the game-graph G that highlights this remark:

  - for all $z \in \mathsf{Out}_{[n]}$ and $v', v'' \in \mathsf{end}(\tau) E_a$, if $z \in \mathsf{In}_i \cap \mathsf{Out}_j$ for some $i, j \in [n]$, then $v'^{\downarrow i \downarrow z} = v''^{\downarrow i \downarrow z} = a^{\downarrow j \downarrow z}$.
  - for all $z \in \mathsf{Out}_0$, if $z \in \mathsf{In}_i$ for an agent $i \in [n]$ and there is a state $v' \in \mathsf{end}(\tau) E_a$ with $v'^{\downarrow i \downarrow z} = [z = 0]$ (or $[z = 1]$), then there exists a state $v'' \in \mathsf{end}(\tau) E_a$ such that $v''^{\downarrow i \downarrow z} = [z = 1]$ (or $[z = 0]$).

  In view of the above remark, to show the main claim of this part, it suffices to show that for an arbitrary state $v' \in \mathsf{end}(\tau) E_a$ and variable $z \in \mathsf{In}_i \cap \mathsf{Out}_{[n]}$ for some $i \in [n]$, if $i \in N_1$, then $v^{\downarrow i \downarrow z} = v_2^{\downarrow i \downarrow z}$, otherwise (that is, if $i \in [n] \setminus N_1$) we have $v^{\downarrow i \downarrow z} = v_1^{\downarrow i \downarrow z}$. We argue this in the following two steps:

- By definition of the edge relation E of the game graph G, it follows that for every pair of agents $i, j \in N_1$ and variable $z \in \mathsf{In}_i \cap \mathsf{Out}_j$, we have $v'^{\downarrow i \downarrow z} = a^{\downarrow j \downarrow z}$. Similarly, for the state $v_2 \in \mathsf{end}(\tau_2)E_{a_2}$, we have that $v_2^{\downarrow i \downarrow z} = a_2^{\downarrow j \downarrow z}$ for every pair of agents $i, j \in N_1$ and variable $z \in \mathsf{In}_i \cap \mathsf{Out}_j$. Since by construction of a, we have $a^{\downarrow j} = a_2^{\downarrow j}$ for all $j \in N_1$, it follows that $v'^{\downarrow i \downarrow z} = v_2^{\downarrow i \downarrow z}$ for all agents $i, j \in N_1$ and variable $z \in \mathsf{In}_i \cap \mathsf{Out}_j$.

  Symmetrically, we can argue that $v'^{\downarrow i \downarrow z} = v_1^{\downarrow i \downarrow z}$ for all agents $i, j \in [n] \setminus N_1$ and variable $z \in \mathsf{In}_i \cap \mathsf{Out}_j$.

- Next we argue that

  * for every agent $i \in N_1$, agent $j \in [n] \setminus N_1$ and variable $z \in \mathsf{In}_i \cap \mathsf{Out}_j$, we have $v'^{\downarrow i \downarrow z} = v_2^{\downarrow i \downarrow z}$.
  * for every agent $i \in [n] \setminus N_1$, agent $j \in N_1$ and variable $z \in \mathsf{In}_i \cap \mathsf{Out}_j$, we have $v'^{\downarrow i \downarrow z} = v_1^{\downarrow i \downarrow z}$.

  Firstly, consider the first case above and note that since Ar is a (weak) $\mathfrak{N}$-broadcast architecture, we know that every variable $z \in \mathsf{In}_{N_1} \cap \mathsf{Out}_{[n] \setminus N_1}$ is written on by a least informed agent i_c of some partition $N_c \in \mathfrak{N} \setminus \{N_1\}$. Symmetrically, every variable $z \in \mathsf{In}_{[n] \setminus N_1} \cap \mathsf{Out}_{N_1}$ is written on by the least informed agent i_1 of the partition N_1. Therefore in both the cases, the variable $z \in \mathsf{In}_i \cap \mathsf{Out}_j$ being referred to, satisfies $z \in \mathsf{Out}_{i_c}$ for some partition $N_c \in \mathfrak{N}$.

  Next we claim that for any least informed agent i_c of any partition $N_c \in \mathfrak{N}$, and variable $z \in \mathsf{Out}_{N_c} \cap \mathsf{In}_{[n] \setminus N_c}$, we have $a_1^{\downarrow i_c \downarrow z} = a_2^{\downarrow i_c \downarrow z} = a^{\downarrow i_c \downarrow z}$. Put in words, this says that for each partition N_c, a variable z written on by the least informed agent $i_c \in N_c$ and read by some agent outside N_c, the agent i_c assigns the same value to the variable z in both the actions a_1 and a_2. It is not difficult to see that if this were true, then the claim made in this part follows.

To see the above claim, observe that since τ'_1, τ'_2 belong to the same modular-state $\mathscr{M}'$, it follows from the definition of the final modularization $\equiv$ that there exists a sequence $\rho_1 a'_1 v'_1, \rho_2 a'_2 v'_2, .., \rho_k a'_k v'_k$ of histories in $\mathsf{dom}(\mathscr{M}')$ such that: (1) the sequence begins at τ'_1 and ends at τ'_2, and (2) for every index $1 \leq j < k$, there exists an agent i such that $\rho_j a'_j v'_j \approx_i \rho_{j+1} a'_{j+1} v'_{j+1}$. It follows from this and the broadcast property shown in Proposition 3.1.1 that for every index $1 \leq j < k$, index $c \in [d]$ and variable $z \in \mathsf{Out}_{N_c} \cap \mathsf{In}_{[n]\setminus N_c}$, we have $a_j'^{\downarrow i_c \downarrow z} = a_{j+1}'^{\downarrow i_c \downarrow z}$. Since $\rho_1 a'_1 v'_1, \rho_k a'_k v'_k$ are the histories τ'_1 and τ'_2 respectively, it follows from the definition of τ'_1, τ'_2 that $a_1^{\downarrow i_c \downarrow z} = a_2^{\downarrow i_c \downarrow z}$. Furthermore, we have by definition of a that $a_1^{\downarrow i_c \downarrow z} = a_2^{\downarrow i_c \downarrow z} = a^{\downarrow i_c \downarrow z}$.

This completes the proof of the theorem. □

6.3 Retraction of Factorizable Witnesses

Having described the notion of factorization of a witness, we show in this section that the retraction of a witness may be constructed in terms of the retractions of its factors. The usefulness of this idea should be clear for the particular kind of canonical witness $\mathscr{W}$ considered here, since each N_c-factor of the canonical witness $\mathscr{W}$ is a modular witness, and we know that modular witnesses admit good retractions[5]. We note here that to construct good retractions for a witness, it is not sufficient to have good retractions for its factors; the way in which these factors are 'coupled' also plays an important role. To counter this, we give a 'compatibility' condition on the retractions of the various factors, so that retractions for a witness may now be constructed from the retractions of its factors.

We begin with some notations.

- For a function $g : S \to S$ on the witness $\mathscr{S}$, we denote the g-retract of $\mathscr{S}$ by $g(\mathscr{S}) =$

[5]As in the previous chapter, the informal term *good* retraction refers to retractions that transform a MaLTS to a bounded-size MaLTS.

$(S^g, R^g, \{\sim_i^g\}, \{\ell_V^g\}, g(s_\varepsilon))$.

- For any N-factor $\mathscr{S}^N = (S^N, R^N, \{\sim_i^N\}_{i\in N}, \{\ell_{V^N}\}, [s_\varepsilon]^N)$ of the witness $\mathscr{S}$ and function $g^N : S^N \to S^N$, we denote the g^N-*retract* of $\mathscr{S}^N$ by $g^N(\mathscr{S}^N) = (S^{N,g^N}, R^{N,g^N}, \{\sim_i^{N,g^N}\}_{i\in N}, \{\ell_{V^N}^{g^N}\}, g^N([s_\varepsilon]^N))$.

Next we define the notion of retraction of $(\{V^N\}, \{A_i\}_{i\in N})$-MaLTS's, which is similar to the definition of retraction of $(\{V\}, \{A_i\}_{i\in[n]})$-MaLTS's defined in Chapter 4.

A function $g^N : S^N \to S^N$ is called a retraction of an N-factor $\mathscr{S}^N$, if the following hold:

R$'$.1 For any state $[s]^N \in S^{N,g^N}$ we have $\ell_{V^N}([s]^N) = \ell_{V^N}(g^N([s]^N))$,

R.2 for any agent $i \in N$ and transitions $([s_1]^N, a^N, [s_2]^N), ([s_1']^N, a'^N, [s_2']^N) \in R^{N,g^N}$ with $[s_1]^N, [s_1']^N \in S^{N,g^N}$, if $[s_1]^N \sim_i^N [s_1']^N$, then,
$[s_2]^N \sim_i^N [s_2']^N$, if and only if, $\ell_{V^N}([s_2]^N)^{\downarrow i} = \ell_{V^N}([s_2']^N)^{\downarrow i}$ and $a^{N\downarrow i} = a'^{N\downarrow i}$,

R$'$.3 For any g-path $[s_0]^N a_1^N [s_1]^N a_2^N ..$ and agent $i \in N$, the priority sequence
$\gamma_i(\ell_{V^N}([s_0]^N)^{\downarrow i}), \gamma_i(\ell_{V^N}([s_1]^N)^{\downarrow i}), ..$ satisfies the parity condition.

The crucial property of a retraction of an N-factor $\mathscr{S}^N$ is that it preserves the property of the retract being witness-like. This can be argued along lines similar to those of Retraction Theorem **??**, and so we only state the theorem here for reference.

Theorem 6.3.1. *For any $N \subseteq [n]$, if g^N is a retraction of an N-factor $\mathscr{S}^N$ of the witness $\mathscr{S}$, the g^N-retract $g^N(\mathscr{S}^N)$ is witness-like.*

Next we give a criterion for when the retractions on the N_c-factors of witnesses may be combined to give a retraction for the witness $\mathscr{W}$.

Compatibility of Retractions Given a $\mathfrak{N}$-factorization $(\{\mathscr{S}^{N_c}\}_{c\in[d]}, \mathscr{C})$ of the witness $\mathscr{S}$, and an retraction g^{N_c} of the N_c-factor $\mathscr{S}^{N_c}$ for each $c \in [d]$, we say that $(\{\mathscr{S}^{N_c}\}_{c\in[d]}, \mathscr{C})$ is *compatible* with a tuple of retractions $(g^{N_c})_{c\in[d]}$, if

$$\{(g^{N_c}([s]^{N_c}))_{c\in[d]} \mid s \in S\} \subseteq \mathscr{C}.$$

The notion of compatibility can be understood intuitively as follows: For each N_c-factor $\mathscr{S}^{N_c}$, if a retraction g^{N_c} for $\mathscr{S}^{N_c}$ maps the element $[s]^{N_c}$ to the element $g^{N_c}([s]^{N_c})$, then the resulting tuple $(g^{N_c}([s]^{N_c}))_{c\in[d]}$ must 'correspond' to some state $s' \in S$. This requirement is ensured by insisting that $(g^{N_c}([s]^{N_c}))_{c\in[d]} \in \mathscr{C}$ is true.

Next we show that compatibility is sufficient for obtaining retractions for $\mathscr{W}$.

Theorem 6.3.2. *Let $(\{\mathscr{S}^{N_c}\}_{c\in[d]}, \mathscr{C})$ be an $\mathfrak{N}$-factorization of the witness $\mathscr{S}$ and let $\widetilde{g} = (g^{N_c})_{c\in[d]}$ be a tuple such that, for each $c \in [d]$, g^{N_c} is a retraction of $\mathscr{S}^{N_c}$.*

If $(\{\mathscr{S}^{N_c}\}_{c\in[d]}, \mathscr{C})$ is compatible with $\widetilde{g}$, then there is a retraction g of $\mathscr{S}$.

Proof. Let the $\mathfrak{N}$-factorization $(\{\mathscr{S}^{N_c}\}_{c\in[d]}, \mathscr{C})$ of the witness $\mathscr{S}$ be compatible with the tuple of retractions $\widetilde{g}$.

Let $g : S \to S$ be defined as follows: for every state $s \in S$, $g(s)$ is the unique history s' that satisfies $[s']^{N_c} = g^{N_c}([s]^{N_c})$ for each $c \in [d]$. We claim that the desired retraction for the witness $\mathscr{S}$ is given by g.

Towards our main claim, we show that the function g is well-defined, that is, for each state $s \in S$, there exists a unique state s' that satisfies $[s']^{N_c} = g^{N_c}([s]^{N_c})$ for each $c \in [d]$. Towards this, consider a state $s \in S$, and the associated tuple $([s]^{N_c})_{c\in[d]} \in \mathscr{C}$. Now since $(\{\mathscr{S}^{N_c}\}_{c\in[d]}, \mathscr{C})$ is compatible with $\widetilde{g}$, it follows that the tuple $(g^{N_c}([\tau]^{N_c}))_{c\in[d]}$ belongs to $\mathscr{C}$. Again from the definition of $\mathscr{C}$, it follows that there exists a state s' such that $(g^{N_c}([s]^{N_c}))_{c\in[d]} = ([s']^{N_c})_{c\in[d]}$.

In order to see that such a history s' is unique, consider for a contradiction, distinct states $s_1, s_2 \in S$ such that, $[s_1]^i = g^{N_c}([s]^{N_c})^{\downarrow i}$ and $[s_2]^i = g^{N_c}([s]^{N_c})^{\downarrow i}$ hold for all $c \in [d]$ and $i \in N_c$. Now consider the following equalities:

$$\{s_1\} = \bigcap_{i\in[n]} [s_1]^i = \bigcap_{\substack{c\in[d]\\ \text{and}\\ i\in N_c}} [s_1]^{N_c\downarrow i} = \bigcap_{\substack{c\in[d]\\ \text{and}\\ i\in N_c}} g^{N_c}([s]^{N_c})^{\downarrow i} = \bigcap_{\substack{c\in[d]\\ \text{and}\\ i\in N_c}} [s_2]^{N_c\downarrow i} = \bigcap_{i\in[n]} [s_2]^i = \{s_2\}.$$

Here the first and last equalities follow from Proposition 3.2.1, the second and fifth equalities are follow from $\mathfrak{N}$ being a partitioning of $[n]$, and the remaining equalities follow from our assumption. The desired contradiction follows.

In order to show that g is a retraction, we show next that g satisfies the two basic constraints **R′.1,R.2** and parity constraint **R′.3** for being a retraction.

- **R′.1**: Firstly, we need to show that for all states $s \in S$, $\ell_V(s) = \ell_V(g(s))$, or equivalently that, $\ell_V(s)^{\downarrow i} = \ell_V(g(s))^{\downarrow i}$ for all $i \in [n]$.

  Towards this, consider a state $s \in S$ and observe that for each index $c \in [d]$, we know that $\ell_{V^{N_c}}([s]^{N_c}) = \ell_{V^{N_c}}(g^{N_c}([s]^{N_c}))$ holds by definition of N_c-retraction g^{N_c}. Now consider an index $c \in [d]$, and the following equivalences:

  $$\ell_{V^{N_c}}([s]^{N_c}) = \ell_{V^{N_c}}(g^{N_c}([s]^{N_c}))$$

  $$\Leftrightarrow \ell_{V^{N_c}}([s]^{N_c}) = \ell_{V^{N_c}}([g(s)]^{N_c}) \qquad \text{(By definition of } g\text{)}$$

  $$\Leftrightarrow \ell_V(s)^{\downarrow i} = \ell_V(g(s))^{\downarrow i} \text{ for all } i \in N_c \quad \text{(By definition of } \ell_{V^{N_c}}\text{)}$$

- **R.2**: Next we need to show that for any agent $i \in [n]$ and transitions $(s_1, a, s_2), (s_1', a', s_2') \in R^g$, if $s_1 \approx_i s_2$, then $s_1' \sim_i s_2'$, if and only if, $a^{\downarrow i} = a'^{\downarrow i}$ and $\ell_V(s_1')^{\downarrow i} = \ell_V(s_2')^{\downarrow i}$.

  Towards this, consider an index $c \in [d]$ and agent $i \in N_c$, and consider transitions $(s_1, a, s_2), (s_1', a', s_2') \in R^g$ such that $s_1 \sim_i s_2$. Let $a^{N_c}, a'^{N_c} \in A^{N_c}$ be such that $a^{N_c \downarrow i} = a^{\downarrow i}, a'^{N_c \downarrow i} = a'^{\downarrow i}$ for all $i \in N_c$.

  We begin with the following lemma.

  Lemma 6.3.3. *If $(s_1, a, s_2) \in R^g$, then for any $c \in [d]$, we have $([s_1]^{N_c}, a^{N_c}, [s_2]^{N_c}) \in R^{N_c, g^{N_c}}$ with $a^{N_c \downarrow i} = a^{\downarrow i}$ for all $i \in N_c$.*

  Proof. To see this, firstly observe that by definition of R^g, there exists a $s \in H$ such that $(s_1, a, s) \in M$ and $g(s) = s_2$. Now if $(s_1, a, s) \in R$, then it follows from the definition of the N_c-factor $\mathscr{S}^{N_c}$ that there exist a transition $([s_1]^{N_c}, a^{N_c}, [s]^{N_c}) \in R^{N_c}$. From the definition of g, it also follows that $g^{N_c}([s]^{N_c}) = [g(s)]^{N_c}$. Now since

$([s]^{N_c}, a^{N_c}, [s]^{N_c}) \in R^{N_c}$ and $g^{N_c}([s]^{N_c}) = [g(s)]^{N_c} = [s_2]^{N_c}$ and since $R^{N_c,g^{N_c}}$ is a 'composition' of the relation R^{N_c} with the N_c-retraction g^{N_c}, it follows from the definition of $R^{N_c,g^{N_c}}$ that $([s_1]^{N_c}, a^{N_c}, [s_2]^{N_c}) \in R^{N_c,g^{N_c}}$. $\square$

Since $(s_1,a,s_2),(s'_1,a',s'_2) \in M^g$, it follows from the above claim that there exist transitions $([s_1]^{N_c}, a^{N_c}, [s_2]^{N_c}), ([s'_1]^{N_c}, a'^{N_c}, [s'_2]^{N_c}) \in R^{N_c,g^{N_c}}$. From the basic constraint **R.2** satisfied by the retraction g^{N_c} of the N_c-factor, it follows that if $[s_1]^{N_c} \approx_i^{N_c} [\tau_2]^{N_c}$, then $[\tau'_1]^{N_c} \approx_i^{N_c} [\tau'_2]^{N_c}$, if and only if, $\ell_{V^{N_c}}([\tau_1]^{N_c})^{\downarrow i} = \ell_{V^{N_c}}([\tau_2]^{N_c})^{\downarrow i}$ and $a^{N_c \downarrow i} = a'^{N_c \downarrow i}$.

The desired claim follows from this, since the predicates $s_1 \sim_i s_2$ and $s'_1 \sim_i s'_2$ are equivalent to $[s_1]^{N_c} \sim_i^{N_c} [s_2]^{N_c}$ and $[s'_1]^{N_c} \sim_i^{N_c} [\tau'_2]^{N_c}$ respectively.

- $\mathbf{R'.3}$: Lastly, we need to show that for any g-path $s_0a_1s_1a_2s_2...$ and agent $i \in [n]$, the priority sequence $\gamma_i(\ell_V(s_0)^{\downarrow i}), \gamma_i(\ell_V(s_1)^{\downarrow i}), \gamma_i(\ell_V(s_2)^{\downarrow i}),..$ satisfies the parity condition.

  Now consider a g-path $s_0a_1s_1a_2s_2...$ in $\mathscr{S}$, a partition $N_c \in \mathfrak{N}$ and an agent $i \in [n]$. Now note that by the definition of the g-path $s_0a_1s_1a_2s_2...$, it is the case that $(\tau_k, a_{k+1}, \tau_{k+1}) \in M^g$ for all $k \geq 0$. It follows from this and lemma 3.3.3 that there exists a g^{N_c}-path $[\tau_0]^{N_c}a_1^{N_c}[\tau_1]^{N_c}a_2^{N_c}[\tau_2]^{N_c}...$ in the N_c-factor $\mathscr{W}^{N_c}$, for some $a_1^{N_c}, a_2^{N_c},...$ Now it follows from the definition of the retraction g^{N_c} that the priority sequence $\gamma_i(\ell_{V^{N_c}}([s_0]^{N_c})^{\downarrow i}), \gamma_i(\ell_{V^{N_c}}([s_1]^{N_c})^{\downarrow i}), \gamma_i(\ell_{V^{N_c}}([s_2]^{N_c})^{\downarrow i})...$ satisfies the parity condition. By definition of $\ell_{V^{N_c}}$, we have $\ell_{V^{N_c}}([s]^{N_c})^{\downarrow i} = \ell_V(s)^{\downarrow i}$ for all $s \in S$, it follows that $\gamma_i(\ell_V(s_0)^{\downarrow i}), \gamma_i(\ell_V(s_1)^{\downarrow i}), \gamma_i(\ell_V(\tau_2)^{\downarrow i}),..$ satisfies the parity condition.

This completes the proof of the theorem. $\square$

Next theorem gives a certain relationship between retraction of a witness, retraction of its factors and modularizations of witnesses and its factors. We begin with some terminology.

Consider a $\mathfrak{N}$-factorization $(\{\mathscr{S}^{N_c}\}_{c\in[d]}, \mathscr{C})$ of the witness $\mathscr{S}$. We say that

- the tuple of retractions $(g^{N_c})_{c\in[d]}$ is a $\mathfrak{N}$-*factorization* of the retraction g of the witness $\mathscr{S}$, if for each $c \in [d]$, g^{N_c} is a retraction of the N_c-factor $\mathscr{S}^{N_c}$ such that for all $s \in S$, $g^{N_c}([s]^{N_c}) = [g(s)]^{N_c}$.

- the tuple of modularizations $(\equiv^c)_{c\in[d]}$ is a $\mathfrak{N}$-*factorization* of the modularization $\equiv$ of the witness $\mathscr{S}$, if for each $c \in [d]$, $\equiv^c$ is a modularization of the N_c-factor $\mathscr{S}^{N_c}$ such that for any states $s_1, s_2 \in S$, $[s_1]^{N_c} \equiv^c [s_2]^{N_c}$, if and only if, $s_1 \equiv s_2$.

Theorem 6.3.4. *Let $(\{\mathscr{S}^{N_c}\}_{c\in[d]}, \mathscr{C}_{\mathfrak{N}})$ be the $\mathfrak{N}$-factorization of the witness $\mathscr{S}$ and let the tuples $(g^{N_c})_{c\in[d]}$, $(\equiv^c)_{c\in[d]}$ be the $\mathfrak{N}$-factorization of the retraction g and modularization $\equiv$ of the witness $\mathscr{S}$.*

1. *If for each $c \in [d]$, the modularization $\equiv^c$ is w_c-wide, then $\equiv$ is $w_1.w_2...w_d$-wide.*

2. *If for each $c \in [d]$, the equivalence relation $\equiv^{c,g^{N_c}} = \equiv^c \cap S^{N_c,g^{N_c}} \times S^{N_c,g^{N_c}}$ is a w_c-wide modularization of the g^{N_c}-retract $g^{N_c}(\mathscr{S}^{N_c})$, then $\equiv^g = \equiv \cap S^g \times S^g$ is a $w_1.w_2...w_d$-wide modularization of the g-retract $g(\mathscr{S})$.*

Proof. Towards the first item, assume that the modularization $\equiv^c$ is w_c-wide, for each $c \in [d]$, that is, each equivalence class in $S^{N_c}/\equiv^c$ is of size at most w_c. Now consider an arbitrary equivalence class $S' \in S/\equiv$. We need to show that S' has size at most $w_1.w_2...w_d$.

To see this, firstly consider the set $S'^{N_c} := \{[s]^{N_c} \mid s \in S'\}$, for each $c \in [d]$. By definition of $\equiv^c$, it follows that $S'^{N_c} \in S^{N_c}/\equiv^c$ holds for each $c \in [d]$, and therefore by our above assumption about $\equiv^c$ being w_c-wide, it follows that $|S'^{N_c}| \leq w_c$. Secondly note that by the construction of each S'^{N_c}, it follows that $\{([s]^{N_c})_{c\in[d]} \mid s \in S'\} \subseteq S'_{N_1} \times S'_{N_2} \times .. \times S'_{N_d}$. Moreover due to Proposition 3.2.1, no two states $s_1, s_2 \in S'$ satisfy $([s_1]^{N_c})_{c\in[d]} = ([s_2]^{N_c})_{c\in[d]}$. It follows from these observations that $|S'| \leq w_1.w_2...w_d$.

Towards the second item, for each $N_c \in \mathfrak{N}$, let $\mathscr{S}^{g,N_c}$ denote the N_c-factor of $g(\mathscr{S})$ and let S^{g,N_c} denote its domain. Now observe that for each $c \in [d]$, there is a one-one correspondence between an element of $\mathscr{S}^{g,N_c}$ and those of $g^{N_c}(\mathscr{S}^{N_c})$ which can be described as follows: $S^{g,N_c} := \{[s]^{N_c} \cap S^g \mid [s]^{N_c} \in S^{N_c,g^{N_c}}\}$. This can be proved by an easy induction of the length of the smallest g^{N_c}-path that reaches a state $S^{N_c,g^{N_c}}$.

Now let $(\equiv^{g,N_c})_{c\in[d]}$ be a $\mathfrak{N}$-factorization of the modularization $\equiv^g$. By the one-one correspondence above and the definition of $\mathfrak{N}$-factorization of a modularization, it follows for each $c \in [d]$, that the modularization $\equiv^{g,N_c}$ is w_c-wide due to the modularization $\equiv^{c,g^{N_c}}$ being w_c-wide. Now from a simple application of the first item of the theorem, it follows that $\equiv^g$ is a $w_1.w_2...w_d$-wide modularization of the g-retract $g(\mathscr{S})$.

$\square$

As mentioned in the beginning of this section, the construction of the retraction for the witness $\mathscr{W}$ proceeds by first constructing for each $N_c \in \mathfrak{N}$, a good retraction of its N_c-factor (which we know to be modular MaLTS's), and then combining the retractions of the various factors to obtain the retraction for the witness $\mathscr{W}$. The above theorems give us enough tools to show the second part of the mentioned plan. Next we state some results towards the first part of the plan, which is to construct good retractions for the modular N-factors of witnesses

6.3.1 Retractions of Modular Witnesses and N-Factors

In order to distinguish the notion of witness in Chapter 5 (that is, witness for global winning conditions) from the notion of witness here (that is, witness for local winning conditions), we use the term *global* witness for the former and *local* witness for the latter.

Much like the case of modular global witnesses in Chapter 5, we can also construct good retractions for modular local witnesses and modular N-factors of local witnesses. We point out only the changes needed in the arguments presented in Chapter 5 because of the witness being a local witness. Next we present the main theorem regarding good retractions for local witnesses.

Theorem 6.3.5.

1. *If $\mathscr{S}$ is w-simple local witness, then there exist a retraction g such that the g-retract $g(\mathscr{S})$ has a domain of size $2^{\mathcal{O}(|N|.w^2\ log(|V|.|P|))}$.*

2. *If the N-factor $\mathscr{S}^N$ of a local witness $\mathscr{S}$ admits a (w,m,d)-decomposition $\equiv_1, \dots, \equiv_d$, then there exists a retraction g^N such that, the modularization $\equiv_d^{g^N} = \equiv_d \cap S^{N,g^N} \times S^{N,g^N}$ of the g^N-retract $g^N(\mathscr{S}^N)$ is w^d-wide, where w_d is recursively defined as follows:*

$$w_d := \begin{cases} 2^{\mathcal{O}(|N|.w_{d-1}^2\ log(m.|V|.|P|)}, & \text{if } d > 1 \\ w, & \text{if } d = 1. \end{cases}$$

Note that since $\equiv_d$ is a final modularization of $\mathscr{S}^N$, it follows that $\equiv_d^{g^N}$ is a final modularization of $g^N(\mathscr{S}^N)$ and therefore $g^N(\mathscr{S}^N)$ is a w_d-simple witness.

The items 1 and 2 of the above theorem are analogues of Theorems 2.4.3 and 2.3.10, modified to the case local witnesses and N-factors of local witnesses respectively. Below we give an outline of the correctness of the analogue of Theorem 2.4.3, since the second claim follows from similar arguments.

To begin with, firstly observe that in the theorem 2.4.3, the only properties of a global witness that are being used in their proofs are the Indistinguishability condition and the $\mathscr{Q}$-Parity condition. While local witnesses satisfy the Indistinguishability condition, they satisfy the γ-Parity condition instead of the $\mathscr{Q}$-Parity condition. The only crucial dependence of the particular formulation of the $\mathscr{Q}$-Parity condition is for defining the modular progress measure.

Recall that modular progress measures were constructed to maintain the $\mathscr{Q}$-Parity condition on the retracts of global witnesses. We can use a similar notion to maintain the γ_i-Parity condition on the retracts of local witnesses. The crucial modifications for this is the deterministic automaton $\mathscr{A}_L$ used in the definition of modular progress measure. Previously, the role of $\mathscr{A}_L$ was to provide a labelling of the modular transition system so that all paths 'contained in' a modular path satisfy the $\mathscr{Q}$-Parity condition. Its role now is to ensure that paths 'contained in' a modular path satisfy the γ_i-Parity condition. The construction of this is similar to that in Proposition 2.3.8, with the exception that now the automaton $\mathscr{A}'$ in the proof of Proposition 2.3.8 (the one that accepts the complement of the language L) first guesses the an agent $i \in [n]$ and then checks if some 'thread' violates the parity condition due to the labelling γ_i. For this idea to follow through, the notion of 'thread' needs to be modified appropriately to record each of the priorities given by priority labellings $\{\gamma_i\}_{i\in[n]}$. Barring this, the remaining construction of the $\equiv$-modular progress measure and all the arguments proceed as before.

6.3.2 Retraction for Games of Broadcast Architectures

We are now in a position to show that canonical (deterministic) witnesses that corresponds to (weak) broadcast architectures admit good retractions.

We begin with some assumptions. Let Ar be a (weak) $\mathfrak{N}$-broadcast architecture and let G be the game graph corresponding to Ar. Recall that $\mathscr{W}$ is a witness for game $(G, \{\gamma_i\}_{i\in[n]})$. Consider a sharp final modularization $\equiv$ of the witness $\mathscr{W}$, the $\mathfrak{N}$-factorization $(\{\mathscr{W}^{N_c}\}_{c\in[d]}, \mathscr{C})$ of the witness $\mathscr{W}$ and the $\mathfrak{N}$-factorization $(\equiv^c)_{c\in[d]}$ of the modularization $\equiv$.

The construction of the desired retraction g of $\mathscr{W}$ proceeds in the following steps:

Modularizing the Factors For each $c \in [d]$, consider the $(1,1,|N_c|+2)$-decomposition $\equiv^c_0 \subseteq \equiv^c_1 \subseteq \, .. \, \subseteq \equiv^c_{|N_c|} \subseteq \equiv^c$ of the N_c-factor $\mathscr{W}^{N_c}$, where $\equiv^c_0 \subseteq \equiv^c_1 \subseteq \, .. \, \subseteq \equiv^c_{|N_c|}$ is a sharp $(1,1,|N_c|+1)$-decomposition of the N_c-factor $\mathscr{W}^{N_c}$.

To see that such a decomposition of $\mathscr{W}$ exists, firstly note that by Theorem 3.2.3, there exists a sharp $(1,1,|N_c|+1)$-decomposition of $\mathscr{W}^{N_c}$. Secondly, for any histories $\tau_1, \tau_2 \in H$, if $[\tau_1]^{N_c} \approx_i^{N_c} [\tau_2]^{N_c}$ holds for an agent $i \in N_c$, then $\tau_1 \approx_i \tau_2$ holds as well. Now since $\equiv$

is a final modularization, it follows that $\approx_i \subseteq \equiv$ for any $i \in N_c$, and therefore $\equiv^c$ is final modularization of $\mathscr{W}^{N_c}$. Since $\equiv^c_{N_c}$ is a sharp final modularization, it follows that $\equiv^c_{N_c} \subseteq \equiv^c$.

Retractions of the Modular Factors For each $c \in [d]$, consider a retraction g^{N_c} of $\mathscr{W}^{N_c}$ such that $g^{N_c}(\mathscr{W}^{N_c})$ is a w_{n_c}-simple MaLTS such that $g^{N_c}([\tau]^{N_c}) \equiv^c [\tau]^{N_c}$ for all $[\tau]^{N_c} \in H^{N_c}$, $n_c = |N_c| + 2$ and the number w_{n_c} is recursively defined as follows:

$$w_{n_c} := \begin{cases} 2^{\mathscr{O}(n_c.w^2_{n_c-1}\ log(m.|V|.|P|)}, & \text{if } d > 1 \\ \\ 1, & \text{if } d = 1. \end{cases}.$$

By item 2 of Theorem 3.3.5 such a retraction g^{N_c} exists for each N_c-factor $\mathscr{W}^{N_c}$.

Retraction of Factors to Retraction of Witness Consider the retraction g of the witness $\mathscr{W}$ such that the witness $g(\mathscr{W})$ is a $w_{n_1}.w_{n_2}...w_{n_d}$-simple witness.

We argue that the existence of such a retraction g in two steps. First we argue the following:

Lemma 6.3.6. *$(g^{N_c})_{c\in[d]}$ is compatible with the coupling relation $\mathscr{C}$.*

Proof. To see this, consider an arbitrary history $\tau \in H$, and let $\tau_c \in H$ be a history such that $[\tau_c]^{N_c} = g^{N_c}([\tau]^{N_c})$ for each index $c \in [d]$. To show that $(g^{N_c})_{c\in[d]}$ is compatible with the coupling relation $\mathscr{C}$, it suffices to show that $([\tau_c]^{N_c})_{c\in[d]} \in \mathscr{C}$.

To see this, consider an index $c \in [d]$ and note that by definition of g^{N_c} it follows that $[\tau]^{N_c} \equiv^c [\tau_c]^{N_c}$. Moreover, by definition of $\equiv^c$ it follows that $\tau \equiv \tau_c$. Therefore we have that $\tau \equiv \tau_1 \equiv ... \equiv \tau_c$.

Now consider the unique $\equiv$-modular state $\mathscr{M}$ that contains the histories $\tau, \tau_1, ..., \tau_c$. By Theorem 3.2.4 we know that $\prod_{c\in[d]} \bigcup_{\tau\in\mathsf{dom}(\mathscr{M})} [\tau]^{N_c} \subseteq \mathscr{C}$. Therefore, it follows that the tuples $([\tau]^{N_c})_{c\in[d]}$ and $([\tau_c]^{N_c})_{c\in[d]}$ belong to the coupling relation $\mathscr{C}$. □

Now by Theorem 3.3.2 it follows that there exists a retraction g for the witness $\mathscr{W}$. Moreover, it is also easy to verify that the retraction g constructed in the proof of Theorem

3.3.2 admits a $\mathfrak{N}$-factorization into $(g^{N_c})_{c\in[d]}$.

Next, since each $g^{N_c}(\mathscr{W}^{N_c})$ is a w_{n_c}-simple MaLTS, it follows by Theorem 3.3.4 that the g-retract $g(\mathscr{W})$ is $w_{n_1}.w_{n_2}...w_{n_d}$-simple witness.

Retraction of w-Simple Witness into Bounded Witness Finally, by item 1 of Theorem 3.3.5 it follows that there exists a retraction g' of the witness $g(\mathscr{W})$ such that $g'(g(\mathscr{W}))$ has a domain of size $2^{\mathscr{O}(|N|.w^2\ log(|V|.|P|))}$ where $w = w_{n_1}.w_{n_2}...w_{n_d}$.

This shows that any canonical witness $\mathscr{W}$ of the game $(G, \{\gamma_i\}_{i\in[n]})$, where G is a game corresponding to a broadcast architecture, admits a retraction into a bounded-size witness.

In view of the above observations, the Solvability Theorem **??** on games and the retraction criterion mentioned in Chapter 4, we have the following result:

Theorem 6.3.7. *The (deterministic) distributed synthesis problem for (weak-) broadcast architecture with local specifications is solvable.*

Chapter 7

Concluding Remarks

We have analyzed the distributed synthesis problem by means of a winning strategy problem on imperfect information games. The common feature in the solutions of all the games studied here is that the solution depends entirely on finding nice information patterns in the game tree or the strategy tree. In Chapter 3 we saw that in the case of games with recurring common knowledge of state, the particular information structure generated by the game may be seen as a repeated games where each round consists of playing a bounded duration imperfect information game, and we use this fact to solve the game.

To further explore the connection between information structure of the game and its solvablity, in Chapter 4 we devise a methodology called the *retraction approach*, that considers the information structure of the game as a first-class object. Essentially, we define *witnesses* to the existence of a winning strategy where the witness incorporates in it the information structure of the game. Furthermore, we hypothesise the reason for solvability of games as being due to the existence of small witnesses. For obtaining small witnesses, we set up a generic operation called *retraction* and identify two constraints, called basic and priority constraints, as the key properties required for the existence of a retraction. In the Chapters 5 and 6 we use the methodology developed here to show the solvability of various games.

In Chapter 5 we use the retraction approach to solve the distributed synthesis problem for architectures with (weak) informedness ordering, by first showing that games corresponding to these architectures have no (uniform-distributed) fork-triples, and then showing that witnesses of such games belong to the class of *modular witnesses* and admit good retractions. The key tools introduced here are that of *modularization* of a witness and *modular progress measure*.

In Chapter 6 we use the retraction approach to solve the distributed synthesis problem for various classes of broadcast architectures. Here we attribute two properties, namely *broadcast property* and *grid property*, to the games corresponding to broadcast architectures and show that witnesses of such games admit good retractions. The key technique used here is that of *factorizing* the witness, constructing retractions on the factors and then combining the retractions of the factors to obtain a retraction of the witness.

As stated in the introduction, our ambition is to find a general methodology that provides a fundamental point of view for the winning strategy problem. Looking back at the work here, we believe that the solutions using the retraction approach offers many qualities that we look for, but they also have certain downsides.

On the upside the retraction approach is robust: for example, the deterministic and non-deterministic winning strategy problem could be solved in a single go. Also the method allows analysis for any slight change in the problem statement: for example, the case of games with multiple-start state, modelling delays in communication etc. Another aspect were we are hopeful of the usefulness of the retraction approach, is when the communication between the agents is by means of mechanisms other than through shared variables..

On the downside, the application of the retraction approach in the two cases studied here, require us to introduce complex terminology and many new objects to solve the problem. Another aspect of the two cases studied here is that many arguments and ideas were repeated, but without a clear way to unify them. We believe that this is because the language that we chose to describe the ideas needs enriching. For example, it seems likely that the notions of modularization and factorization, both of which identify sub-structures of a witness, can be better described by other uniform means.

In spite of these downsides, we believe that solutions by means of retraction approach is a step in the right direction, and that explicitly representing the informational changes in a distributed system will be beneficial in the long run. In view of this, we believe that the immediate priority is to find a language that is well-suited to describe analysis by means of the retraction approach.

Bibliography

[1] Gary L. Peterson and John H. Reif. Multiple-Person Alternation. In *Proceedings of the 20th Annual Symposium on Foundations of Computer Science, (FOCS 1979)*, pages 348–363. IEEE, 1979.

[2] Amir Pnueli and Roni Rosner. On the synthesis of a reactive module. In *Proceedings of the 16th Annual ACM Symposium on Principles of Programming Languages (POPL 1989)*, pages 179–190. ACM Press, 1989.

[3] Orna Kupferman and Moshe Y. Vardi. Weak alternating automata are not that weak. *ACM Transactions on Computional Logic*, 2(3):408–429, 2001.

[4] B. Finkbeiner and S. Schewe. Uniform distributed synthesis. In *Proceedings of the 20th IEEE Symposium on Logic in Computer Science (LICS 2005)*, pages 321–330. IEEE, 2005.

[5] P. Madhusudan and P.S. Thiagarajan. Distributed controller synthesis for local specifications. In *Proceedings of the 28th International Colloquium on Automata, Languages and Programming (ICALP 2001)*, volume 2076 of *LNCS*, pages 396–407. Springer, 2001.

[6] Wladimir Fridman and Bernd Puchala. Distributed synthesis for regular and contextfree specifications. *Acta Inf.*, 51(3-4):221–260, 2014. URL `https://doi.org/10.1007/s00236-014-0194-x`.

[7] H. G. Rice. Classes of recursively enumerable sets and their decision problems. *Transactions of the American Mathematical Society*, 74(2):358–366, 1953. ISSN 00029947. URL `http://www.jstor.org/stable/1990888`.

[8] Bernd Finkbeiner and Sven Schewe. *Synthesis of Asynchronous Systems*, pages 127–142. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007. ISBN 978-3-540-71410-1. URL `https://doi.org/10.1007/978-3-540-71410-1_10`.

[9] Anca Muscholl and Igor Walukiewicz. Distributed synthesis for acyclic architectures. In *Proceedings of Foundation of Software Technology and Theoretical Computer Science (FSTTCS 2014)*, volume 29 of *LIPIcs*, pages 639–651. Schloss Dagstuhl, 2014.

[10] Alonzo Church. Application of recursive arithmetic to the problem of circuit synthesis. *Journal of Symbolic Logic*, 28(4):289–290, 1963.

[11] J. Richard Büchi and Lawrence H. Landweber. Solving sequential conditions by finite-state strategies. *Transactions of the American Mathematical Society*, 138: 295–311, 1969.

[12] Michael O. Rabin. Automata on infinite objects and Church's problem. *American Mathematical Society*, 1972.

[13] John H. Reif. Universal games of incomplete information. In *Proceedings of the 11th Annual ACM Symposium on Theory of Computing, STOC '79*, pages 288–308. ACM Press, 1979.

[14] John H. Reif. The complexity of two-player games of incomplete information. *Journal of Computer and Systems Sciences*, 29(2):274–301, 1984.

[15] Orna Kupferman and Moshe Y. Vardi. Church's problem revisited. *The Bulletin of Symbolic Logic*, 5(2):245–263, 1999.

[16] Dietmar Berwanger, Łukasz Kaiser, and Bernd Puchala. A perfect-information construction for coordination in games. In *Proceedings of Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2011)*, volume 13 of *LIPICS*, pages 387–398. Leibniz-Zentrum für Informatik, December 2011.

[17] E. Grädel, W. Thomas, and T. Wilke, editors. *Automata, Logics, and Infinite Games*. Number 2500 in LNCS. Springer-Verlag, 2002.

[18] Nir Piterman. From nondeterministic büchi and streett automata to deterministic parity automata. *Logical Methods in Computer Science*, 3(3), 2007. URL `https://doi.org/10.2168/LMCS-3(3:5)2007`.

[19] Paul J. Krasucki and R. Ramanujam. Knowledge and the ordering of events in distributed systems: Extended abstract. In *Proceedings of the 5th Conference on Theoretical Aspects of Reasoning About Knowledge*, TARK '94, pages 267–283. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1994. ISBN 1-55860-331-X. URL `http://dl.acm.org/citation.cfm?id=1028104.1028124`.

[20] R. Ramanujam. Local knowledge assertions in a changing world: Extended abstract. In *Proceedings of the 6th Conference on Theoretical Aspects of Rationality and Knowledge*, TARK '96, pages 1–14. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1996. ISBN 1-55860-417-9. URL `http://dl.acm.org/citation.cfm?id=1029693.1029694`.

[21] David Janin and Igor Walukiewicz. *Automata for the modal μ-calculus and related results*, pages 552–562. Springer Berlin Heidelberg, Berlin, Heidelberg, 1995. ISBN 978-3-540-44768-9. URL `https://doi.org/10.1007/3-540-60246-1_160`.

[22] David E. Muller and Paul E. Schupp. Alternating automata on infinite trees. *Theoretical Computer Science*, 54:267–276, 1987.

[23] David E. Muller and Paul E. Schupp. Simulating alternating tree automata by nondeterministic automata: New results and new proofs of the theorems of Rabin, McNaughton and Safra. *Theoretical Computer Science*, 141(1–2):69–107, 1995.

[24] Rajeev Alur, Thomas A. Henzinger, and Orna Kupferman. Alternating-time temporal logic. *Journal of the ACM*, 49(5):672–713, 2002.

[25] Bernd Finkbeiner and Sven Schewe. *Distributed Synthesis for Alternating-Time Logics*, pages 268–283. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007. ISBN 978-3-540-75596-8. URL `https://doi.org/10.1007/978-3-540-75596-8_20`.

[26] Dietmar Berwanger and Anup Basil Mathew. Games with recurring certainty. In *Proceedings 2nd International Workshop on Strategic Reasoning, (SR 2014)*, volume 146 of *EPTCS*, pages 91–96, 2014.

[27] Dietmar Berwanger and Anup Basil Mathew. Infinite games with finite knowledge gaps. *Inf. Comput.*, 254:217–237, 2017. URL `https://doi.org/10.1016/j.ic.2016.10.009`.

[28] Moshe Y. Vardi and Pierre Wolper. Reasoning about infinite computations. *Information and Computation*, 115:1–37, 1994.

[29] Neil D. Jones. Space-Bounded Reducibility among Combinatorial Problems. *Journal of Computer and System Sciences*, 11(1):68–85, 1975.

[30] Salman Azhar, Gary Peterson, and John Reif. Lower bounds for multiplayer noncooperative games of incomplete information. *Journal of Computers and Mathematics with Applications*, 41:957–992, 2001.

[31] E. Allen Emerson and Charanjit S. Jutla. Tree automata, mu-calculus and determinacy (extended abstract). In *Proceedings of the 32nd Annual Symposium on Foundations of Computer Science (FOCS 1991)*, pages 368–377. IEEE Computer Society Press, 1991.

[32] Wiesław Zielonka. Infinite games on finitely coloured graphs with applications to automata on infinite trees. *Theoretical Computer Science*, 200(1–2):135–183, 1998.

[33] Erich Graedel and Igor Walukiewicz. Positional determinacy of games with infinitely many priorities. *Logical Methods in Computer Science*, pages 1–22, 2006. URL `https://hal.archives-ouvertes.fr/hal-00335726`.

[34] Nils Klarlund. Progress measures, immediate determinacy, and a subset construction for tree automata. *Annals of Pure and Applied Logic*, 69(2–3):243–268, 1994.

[35] Dietmar Berwanger, Anup Basil Mathew, and Marie van den Bogaard. Hierarchical information patterns and distributed strategy synthesis. In *Automated Technology for Verification and Analysis - 13th International Symposium, ATVA 2015, Shanghai, China, October 12-15, 2015, Proceedings*, pages 378–393, 2015. URL `https://doi.org/10.1007/978-3-319-24953-7_28`.