Strategies in games: a logic-automata study

S. Ghosh[1] and R. Ramanujam[2]

[1] Indian Statistical Institute
SETS Campus, Chennai 600 113, India.
sujata@isichennai.res.in

[2] The Institute of Mathematical Sciences
C.I.T. Campus, Chennai 600 113, India.
jam@imsc.res.in

1 Introduction

Overview. There is now a growing body of research on formal algorithmic models of social procedures and interactions between rational agents. These models attempt to identify logical elements in our day-to-day social activities. When interactions are modeled as games, reasoning involves analysis of agents' long-term powers for influencing outcomes. Agents devise their respective strategies on how to interact so as to ensure maximal gain. In recent years, researchers have tried to devise logics and models in which strategies are "first class citizens", rather than unspecified means to ensure outcomes. Yet, these cover only basic models, leaving open a range of interesting issues, e.g. communication and coordination between players, especially in games of imperfect information. Game models are also relevant in the context of system design and verification. In this article we will discuss research on logic and automata-theoretic models of games and strategic reasoning in multi-agent systems. We will get acquainted with the basic tools and techniques for this emerging area, and provide pointers to the exciting questions it offers.

Content. This article consists of 5 sections apart from the introduction. An outline of the contents of the other sections is given below.

1. **Section 2 (Exploring structure in strategies):** A short introduction to games in extensive form, and strategies. Discussion of some specific strategies. Introduction to games of unbounded duration, and the motivation for such games in the study of multi-agent systems.

2. **Section 3 (Automata theory for strategies in games):** Backward induction in games on finite graphs with reachability objectives. Going from memoryless strategies to bounded memory strategies for regular objectives. Strategies as finite state automata, and automata models for temporal logics.

3. **Section 4 (Game logic and its descendants):** Rohit Parikh's game logic - a discussion on decidability, non-normality, the role of iteration and dual

operators, an embedding into the mu-calculus; multi-player situations, parallel games, coalitions.

4. **Section 5 (Making strategies explicit):** A short overview of various logical frameworks that talk about strategies explicitly in the language, and then moving on to detailed logical studies of structured strategizing in extensive form games.

5. **Section 6 (Dynamics of large games):** Discussion of various issues regarding the study of strategy structure under perfect and imperfect information in large games, e.g. strategy switching, stabilizing strategies, and others. A preliminary look into dynamic games forms.

Lecture slides on these materials can be found at the followng location:
`http://www.ai.rug.nl/~sujata/documents.html`.

About the authors. *S. Ghosh* is a member of the faculty at the Indian Statistical Institute, Chennai. She works on building up logical frameworks of multi-agent systems using game-theoretic studies and models dynamics of information in such agent-based systems.
R. Ramanujam is a member of the faculty at the Institute of Mathematical Sciences, Chennai. His research interests are in logic and automata theory, and their applications in computer science, especially in distributed and communicating systems, security theory and game theory.

Acknowledgements. The content includes research that has been jointly done with Johan van Benthem, Fenrong Liu, Soumya Paul and Sunil Simon. We gratefully acknowledge their contributions. We also thank Soumya Paul and Sunil Simon for helping us in preparation of the lecture slides associated with the materials. The first author thanks Netherlands Organisation of Scientific Research grant 600.065.120.08N201 for financial support during her stay at the University of Groningen, where this work was undertaken.

2 Exploring structure in strategies

2.1 Unknown player types

Suppose that you are in a crowded room, with perhaps more than 50 persons in it, and everyone in the room participates in a game. You are to pick a real number in the range 0 to 100. The one who comes closest to *two-thirds* of the average gets a prize. What number would you pick? If the game were played repeatedly, would you play the same way each time? If Ms X were to win the prize using number x, would that influence you the next time you played?

Game theory advises, rather insists, that you pick 0. The reasoning is as follows: two-thirds of the average can never exceed 67, so all numbers above 67 are eliminated. Since nobody is going to write numbers above 67, two thirds of the resulting average will never exceed 44. Hence all numbers above 44 are eliminated. And so on.

However, if we actually play the game,[3] we find that the prize goes not to 0 but some number around 20.[4] Why does this happen? One reasonable conjecture is that people reason one level down but not further. But reasons aside, empirical evidence of this kind is often cited to attack a standard assumption in game theory: common knowledge of rationality.

Our motive in considering this game is different. It is to point out that something well known in game theory: strategizing during play is different from reasoning about existence of (optimal) strategies. Played strategies have considerable *structure*, they tend to be heuristic, partial, based on consideration of other player types, evolve in time and are often composed using other partial strategies that work in other game contexts.

In this game, the number you pick is determined by your own reasoning ability, your conception of alternate player types, your expectation of how other player types are distributed and how you revise such expectation when the result is announced. Note that even if the game is repeated, unless the announced result is close to 0, you get little actual information about how player types are distributed.

Apart from heuristic play, a structure theory of strategies can offer mathematical insight as well, and this is illustrated by the next example.

2.2 The Game of Nim

In this game we begin with m heaps, each heap having some number of counters. Two players take turns alternately, a move consisting of picking heap i, and removing a non-zero number of counters in it. A player whose turn it is to move but has no move available, loses; this happens when all heaps are empty.

[3] The second author has done this many times, among university students, researchers, school children. The results are similar.

[4] In ESSLLI 2011 in Ljubljana, the prize – chocolate – went to 18.54.

Proposed and completely solved by Bouton in 1902 [6], the game nevertheless has attracted considerable interest. It is an example of a bipartisan[5] two-player zero-sum game of perfect information, but more than that, every such game can be reduced to Nim [3]. Rather than kill the reader's joy by providing a solution, we merely point out some interesting mathematical structure in the space of Nim strategies, as discussed by [3].

Suppose you are faced with the situation $(1, 1)$. This is a losing position for you, since no matter which heap you reduce (to empty), the opponent can move and then you have no move. Now, $(2, 2)$ is also losing since no matter what move you make on one heap, the opponent can copy that move on the other heap and thus bring you to $(1, 1)$ or the empty configuration. Thus, in general, we can argue that the Nim position (m, n) is winning to a player iff $m \neq n$.

But this means that (k, m, n) is winning when $k = m$, since you can remove the heap with n counters entirely and present a losing position to the opponent. Similarly (m, m, n, n) is losing: we can consider it as consisting of two subgames, one with configuration (m, m) and the other with (n, n), or two subgames, each with (m, n) as configuration. Note that whatever move player I chooses in one subgame, player II has a *copycat* strategy to make the same move in the other subgame and hence cannot lose.

What kind of subgame composition are we speaking of here? Suppose we represent the game (m, m, n, n) as $(m, m) + (n, n)$. Then the move of player I would be to choose one of the two subgames, and then make a move as above. Suppose she moves in (m, m) leading to the position (m, k). Since player II can now move again in either of the two subgames we have the game $(m, k) + (n, n)$.

In general, if a move in game g can lead to configurations $g_1, \ldots, g_m$ and game h can lead to $h_1, \ldots, h_n$ we can define the **sum** game $g + h$ as that game which can lead to configurations $g_1 + h, \ldots, g_m + h, g + h_1, \ldots, g + h_n$. That is, the player (whose turn it is to play) either moves in g or in h. From now on we speak of a game as winning or losing game to mean that it is winning or losing for the player whose turn it is to move.

Let 0 denote the game with the null heap. Clearly this is a losing game. As expected, $g + 0 = g$ for any game g. It is easy to see that $+$ is commutative and associative. Is there more algebraic structure? Can one equate games g and h based on availability of strategies? Yes, but this requires a notion of game equivalence.

Definition 2.1. *Two games g, g' are said to be* **equivalent***, written $g \equiv g'$ if and only if for any other game h, the sum $g + h$ is winning if and only if $g' + h$ is winning.*

It is easy to verify that $\equiv$ is indeed an equivalence relation. We can follow the definition carefully and show that:

Proposition 2.1. *If g is a losing game, then $g \equiv 0$.*

[5] A game in which both players have identical sets of moves.

Therefore a losing game can be added to any game without changing its win/lose status. This is a simple observation that can be immensely useful. For example, $(1, 2, 3)$ is losing, since any move leads to the form (m, n) or (m, m, n) with $m \neq n$ which, we have seen, is winning. Also $(4, 5)$ is winning, and by the proposition above, $(1, 2, 3, 4, 5)$ is winning, which is hard to prove directly. Thus exploiting structure in strategies simplifies analysis greatly.

How much structure can we actually find? The next proposition offers a clue.

Proposition 2.2. (The copycat principle)*: For any Nim game g, we have: $g + g \equiv 0$.*

This is easily seen: suppose player I has a move leading to $g' + g$. Then player II can mimic with the same move in the other game leading to $g' + g'$, which is losing by subgame induction, and hence $g + g$ is losing as well.

Thus every game acts as its own additive inverse, and yes, Nim games form a **group** under the sum operation, with every element acting as its own inverse.

2.3 Exploring structure

We have merely touched on algebraic structure in combinatorial games; the so-called *Sprague - Grundy theory* has much more to offer [3].

But extensive form games in general exhibit such structure as well. We saw that the analysis above proceeded by considering subgames and how player moves result in one subgame or another. Given a Nim configuration, we can draw the tree of possible configurations obtained by alternating moves. What we exploited above was the symmetric structure present in subgames, Nim being an impartial game. In some sense, history of moves is irrelevant in playing Nim.

In general, when games have temporal structure, the history of such moves, and how much of history is/can be recorded by players can be important, and this dictates structure in strategies as well. In particular, we can study memory structure: how much memory is needed for a particular strategy. This suggests a realisation of strategies by way of finite state automata (or more precisely, transducers).

Another kind of structure we can consider is *compositional*. In this view, players have a library of simple heuristics that work in some situations but not always. Then they compose these online during play, based on and in response to other players' moves. This is akin to how programs are composed in programming languages. Here we are led to iterative structure and the kind of reasoning employed in dynamic logics.

While these considerations apply to games with large temporal structure, there are also games with large *spatial* structure. These are games with a large number of players, and like in the game for guessing two-thirds of the average, players have only expectations on player type distributions and act on that basis. Strategy structure involves heuristics, neighbourhood switching, and so on.

These are generic, independent of specific games. When we consider games arising from specific classes of applications, these application domains offer further structure as well. We will discuss some of these domains; but our main attempt is not to state (the obvious) that game theory is widely applicable, but that strategies are worth studying as first class citizens in a wide variety of contexts, rather than asserting their mere existence in those contexts.

2.4 Some application domains

To motivate our studies from a broader perspective, we now provide a brief look at some of the application domains. Studying structured strategies might throw some light into understanding the actions, interactions that an individual or a group considers to obtain a certain outcome or to attain stabilization in a system.

Negotiations From mundane talks between partners about who will fetch the children from school and who will cook dinner, through sale of an apartment while the seller is trying to hide from the buyer that she has already bought a new house, to the full-fledged multi-party multi-issue negotiations in Kyoto and Copenhagen about climate control – negotiation is everywhere. It is a complex skill that is not easily learnt, and hence could be broken off quite easily, even when they have potential for a win-win solution. Moreover, in many negotiations that do result in an agreement, it may happen that one or more participants could have done better for themselves [37,9,38].

Consider the negotiations in the second phase of the Strategic Arms Limitation Treaty (SALT) during the 1970's (`http://en.wikipedia.org/wiki/Strategic_Arms_Limitation_Talks`) between the Soviets and the Americans in the cold war era, or, the Camp David negotiations between Israel and Egypt in 1978 with US President Carter acting as a mediator (`http://en.wikipedia.org/wiki/Camp_David_Accords`). These situations provide interesting examples of strategic communication and decision making under imperfect information where composite strategies come into play. We will come back to the discussion on strategic communication in games of imperfect information in Section 6.

Cognitive studies In cognitive science, the term 'strategy' is used much more broadly than in game theory. A well-known example is formed by George Polya's problem solving strategies (understanding the problem, developing a plan for a solution, carrying out the plan, and looking back to see what can be learned) [36]. Nowadays, cognitive scientists construct fine-grained theories about human reasoning strategies [23,21], based on which they construct computational cognitive models. These models can be validated by comparing the model's predicted outcomes to results from experiments with human subjects [1].

Various cognitive scientists conduct behavioral experiments to investigate how well humans are able to apply first and second order reasoning. First order reasoning involves reasoning about first order epistemic attributions, e.g. "You believe that I am holding a red card", whereas second order reasoning involves second order attributions, e.g. "I believe that you believe that I am holding a red card". Researchers present participants with strategic games to investigate higher-order social reasoning [19,25], based on the strategic reasoning applied in such games. Based on these strategic game experiments, computational cognitive models are built to describe human strategic reasoning. It has been proposed how formal models of strategic reasoning could aid in building up such cognitive models based on experimental findings [13,12].

Besides, there are various other interactive situations where such complex strategic reasoning comes into play when the temporal or the spatial structure of the game is large enough. Whether we think of obtaining effective bargaining protocols, bidding in different types of auctions, stabilizing entering firms in existing markets, obtaining cartel agreement among competing firms, effective and efficient strategies are needed everywhere.

The concept of strategies also plays a role in language use and interpretation. For example, pragmatics can be explained in terms of a sender and receiver strategizing to understand and be understood, on the basis of concise and efficient messages. Also, evolutionary game theory has been used to explain the evolution of language; for example, it has been shown that in signaling games, evolutionarily stable states occur when the sender's strategy is a one-one map from events to signals, and the receiver's strategy is the inverse map [42].

With this brief introduction to the various application areas, we will now move on to more technical discussions on 'structured strategic reasoning'.

3 Automata theory for strategies in games

Consider a game where two players take turns as follows. Player I picks a natural number, II responds with a larger number, then I picks a further larger number, and so on. If the resulting infinite play is in a given set $A \subseteq \mathbb{N}^\omega$, then player I wins, otherwise II wins (and I loses). Such two-player zero-sum infinite games of perfect information have a long and rich history.

We say that such a game A is determined if either of the two players has a *winning strategy*. In the 1930's, Ulam asked for a characterization of determined games. It did not take long for people to realize that non-determined games exist. In 1953, the Gale-Stewart theorem [11] established determinacy of open games (open in the Baire topology). This led to a series of theorems culminating in Martin's theorem asserting determinacy of Borel games [24].

In the sequel, we are not interested in infinite games in all their generality, but we do wish to consider games of infinite duration. One reason for this is our consideration of games with large temporal structure, even finite ones, as explained below.

A classic example of such a game is the game of chess. Zermelo showed in [44] that chess is determined, i.e. from every game position, either there exists a (pure) strategy for one of the two players (white or black) guaranteeing that she will win or each one of the two players has a strategy guaranteeing at least a draw. However, given any game position, we do not know which of the three alternatives is the correct one. For games like Hex, it is known that the first player can force a win [10] but nonetheless a winning strategy is not known. Again, in such situations, rather than be content with reasoning *about games* using the functional notion of strategies, one needs to reason *about strategies* themselves. For instance, most of the chess playing programs use heuristics which are basically partially specified strategies. A library of such specifications is developed and during the course of play, the actual strategy is built up by composing various partial strategies. Reasoning here consists of strategy selection and composition, based on local outcomes that they yield.

Crucially, a resource bounded player who reasons locally in a large game like chess does not reason with an unknown finite tree of fixed size, but with one of unbounded size, a potentially infinite tree. We consider such reasoning to be inescapable, in the following sense: when a player reasons in a large finite game, she can only look for repetitive patterns and strategize accordingly, as done in regular infinite games.

Another point raised above is that strategies tend to be partial. A strategy is a function from the set of partial plays to moves: it advises a player at a game position on the choice she can make. In a large game, this amounts to a complete specification of behaviour in all possible game situations. But then in such a game, one player's knowledge of the strategies employed by the other is necessarily partial. Rational play requires much finer analysis since strategies have structure that depends on the player's observations of game positions, history of play and the opponent's apparent strategies. We suggest that study of structure in strategies is relevant even in finite, determined, but large, zero-sum games, such as Chess.

Below, we suggest that standard automata theoretic techniques can be employed to usefully specify and analyze partial strategies in non-zero games on graphs. We propose a syntactic framework for strategies in which best response can be algorithmically determined, and a simple modal logic in which we can reason about such strategies. This proposal is intended more as an illustration of such analysis; ideally, we need a "programming language" for strategies, whose structure should be determined empirically by how well they describe interesting heuristics. Such heuristics have been employed in many classes of games that arise in applications mentioned above.

We consider only finitely-presented infinite games. For this, it is convenient to conceive of the game as played on a finite graph, and the game tree obtained by its unfolding.

3.1 Games on graphs and strategies

We begin with a description of the game arena. We use the graphical model for extensive form turn-based games, where at most one player gets to move at each game position.

Game Arena

Let $N = \{1, 2\}$ be the set of players and $\Sigma = \{a_1, a_2, \ldots, a_m\}$ be a finite set of action symbols, which represent moves of players.

A game arena is a finite graph $\mathcal{G} = (W^1, W^2, \longrightarrow, s_0)$ where W^i is the set of game positions of player i for $i \in N$. Let $W = W^1 \cup W^2$. The transition function $\longrightarrow : (W \times \Sigma) \to W$ is a partial function also called the move function and s_0 is the initial node of the game. Let $\bar{i} = 2$ when $i = 1$ and $\bar{i} = 1$ when $i = 2$.

Let the set of successors of $s \in W$ be defined as $\vec{s} = \{s' \in W \mid s \xrightarrow{a} s'$ for some $a \in \Sigma\}$. We assume that for all game positions s, $\vec{s} \neq \emptyset$. Note that the set of game positions in the arena is finite, but looping would lead to visiting the same positions infinitely often, and strategizing may involve carrying information about returning to a position some number of times.

In an arena, the play of a game can be viewed as placing a token on s_0. If player i owns the game position s_0 (i.e $s_0 \in W^i$), then she picks an action 'a' which is enabled for her at s_0 and moves the token to s' where $s_0 \xrightarrow{a} s'$. The game then continues from s'. Formally, a play in $\mathcal{G}$ is an infinite edge labelled path $\rho : s_0 a_0 s_1 a_1 \cdots$ where $\forall j : s_j \xrightarrow{a_j} s_{j+1}$. Let *Plays* denote the set of all plays in the arena.

Games and Winning Conditions Let $\mathcal{G}$ be an arena as defined above. The arena merely defines the rules about how the game progresses and terminates. More interesting are the winning conditions of the players, which specify the game outcomes. Since we consider non-zero sum games, players' objectives need not be strictly conflicting, and each player has a preference relation inducing an ordering over the set of valid plays. The game is specified by presenting the game arena along with the preference relation for each player. Let $\preceq^i \subseteq (\mathit{Plays} \times \mathit{Plays})$ be a complete, reflexive, transitive binary relation denoting the preference relation of player i for $i \in N$. Then the game G is given as, $G = (\mathcal{G}, \{\preceq^i\}_{i \in N})$.

In general, the preference relation need not have a finite presentation, and we restrict our attention to finite state preferences. (This is because in the applications we have in mind, as in network games, desired or preferred plays are easily expressed as formulas of temporal logics.) Thus, the preferences of players are presented as finite state evaluation automata, with Muller acceptance conditions.

Let $\mathcal{M} = (R, \Delta, r_0)$ be a deterministic automaton with finite set of states R, initial state $r_0 \in R$ and transition function $\Delta : R \times W \times \Sigma \to R$. The evaluation automaton is given by: $\mathcal{E} = (\mathcal{M}, \{\lhd^i\}_{i \in N})$ where $\lhd^i \subseteq (F \times F)$ is a total order over $F = 2^R \setminus \emptyset$ for $i \in N$.

A run of $\mathcal{E}$ on a play $\rho : s_0 a_0 \cdots \in \mathit{Plays}$ is a sequence of states $\varphi : r_0 r_1 \cdots$ such that $\forall i : 0 \leq i < n$, we have $r_{i+1} = \Delta(r_i, s_i, a_i)$. Let $\mathit{inf}(\varphi)$ denote the set of states occurring infinitely often in φ. The evaluation automaton $\mathcal{E}$ induces a preference ordering on Plays in the following manner. Let $\rho : s_0 a_0 s_1 \cdots$ and $\rho' : s_0 a'_0 s'_1 \cdots$ be two plays. Let the run of $\mathcal{E}$ on ρ and ρ' be $\varphi : r_0 r_1 \cdots r_n$ and $\varphi' : r_0 r'_1 \cdots r'_n$ respectively. For $i \in N$, we have $\rho \preceq^i \rho'$ iff $\mathit{inf}(\varphi) \lhd^i \mathit{inf}(\varphi')$. A game is presented as $G = (\mathcal{G}, \mathcal{E})$.

We will also be interested in binary evaluation automata which specify least outcomes for player i. Such an automaton is given by $\mathcal{E}^i_F$, where $F \in 2^R$: for every $F' \in 2^R$, if $F \lhd^i F'$, it is taken to be "winning" for player i, and every $F'' \neq F$ such that $F'' \lhd^i F$ is taken to be "losing". Such an automaton checks if i can ensure an outcome which is at least as preferred as F. Note that the terminology of win/loss is only to indicate a binary preference for player i, and applies even in the context of non-zero sum games.

Thus we have game arenas, with players' preference on plays. We now discuss strategies of players.

Strategies Let $\mathcal{G}_T$ denote the tree unfolding of the arena $\mathcal{G}$. (This is the tree whose root is labelled s_0, the initial game position, and a node labelled s has a child labelled s' iff there is a move at s that leads to position s' in $\mathcal{G}$.) We use s, s' to denote the nodes in $\mathcal{G}_T$. A strategy for player 1, $\mu = (W^1_\mu, W^2_\mu, \longrightarrow_\mu, s_0)$ is a maximal connected subtree of $\mathcal{G}_T$ where for each player 1 node, there is a unique outgoing edge and for the other player every move is included. (We will use $W_\mu = W^1_\mu \cup W^2_\mu$.)

That is, for $s \in W_\mu$ the edge relation satisfies the following property:

- if $s \in W^1_\mu$ then there exists a unique $a \in \Sigma$ such that $s \xrightarrow{a}_\mu s'$.
- if $s \in W^2_\mu$, then for each s' such that $s \xrightarrow{a}_T s'$, we have $s \xrightarrow{a}_\mu s'$.

Let Ω^i denote the set of all strategies of Player i in $\mathcal{G}$, for $i = 1, 2$. We will use μ to denote a strategy of player 1 and τ a strategy of player 2. A strategy profile $\langle \mu, \tau \rangle$ defines a unique path ρ^τ_μ in the arena $\mathcal{G}$.

In games with overlapping objectives, the common solution concept employed is that of an equilibrium strategy profile [27]: a profile of strategies, one for each player, is said to be in equilibrium if no player gains by unilaterally deviating from his strategy. The notion of equilibrium can be formally defined as follows. Let μ denote a strategy of player 1 and τ denote a strategy of player 2.

- μ is the best response for τ iff $\forall \mu' \in \Omega^1$, $\rho^{\tau}_{\mu'} \preceq^1 \rho^{\tau}_{\mu}$.
- τ is the best response for μ iff $\forall \tau' \in \Omega_2$, $\rho^{\tau'}_{\mu} \preceq^2 \rho^{\tau}_{\mu}$.
- $\langle \mu, \tau \rangle$ is a Nash equilibrium iff μ is the best response for τ and τ is the best response for μ.

The natural questions that are of interest include:

- Given a strategy τ of player 2, what is the best response for player 1?
- Given a strategy profile $\langle \mu, \tau \rangle$, is it a Nash equilibrium?
- Does the game possess a Nash equilibrium?

Clearly, if we can answer the first question, we can answer the second as well. In any case, to study these questions algorithmically, we need to be able to present the preferences of players and their strategies in a finite fashion. We have evaluation automata presenting preferences; we now proceed to a syntax for strategies.

3.2 Strategy specification

We conceive of strategies as being built up from atomic ones using some grammar. The atomic case specifies, for a player, what conditions she tests for before making a move. We can associate with the game arena a set of observables for each player. One elegant method then, is to state the conditions to be checked as a past time formula of a simple tense logic over the observables. The structured strategy specifications are then built from atomic ones using connectives. We crucially use an implication of the form: "if the opponent is apparently playing a strategy π then play σ".

Below, for any countable set X, let $Past(X)$ be a set of formulas given by the following syntax:

$$\psi \in Past(X) := x \in X \mid \neg\psi \mid \psi_1 \vee \psi_2 \mid \diamondminus\psi.$$

Syntax Let $P^i = \{p^i_0, p^i_1, \ldots\}$ be a countable set of observables for $i \in \{1,2\}$ and let $P = P^1 \cup P^2$. The syntax of strategy specifications is then given by:

$$\sigma \in Strat^i(P^i) := [\psi \mapsto a]^i \mid \sigma_1 + \sigma_2 \mid \sigma_1 \cdot \sigma_2 \mid \pi \Rightarrow \sigma_1$$

where $a \in \Sigma$, $\pi \in Strat^{\bar{\imath}}(P^1 \cap P^2)$ and $\psi \in Past(P^i)$. The sets P^1 and P^2 need not be disjoint. Observe that since the atomic specifications are always indexed by the player identity, it cannot be the case that a strategy specification $\sigma \in Strat^i(P^i) \cap Strat^{\bar{\imath}}(P^{\bar{\imath}})$.

The idea is to use the above constructs to specify properties of strategies. For instance the interpretation of a player i specification $[p \mapsto a]^i$ where $p \in P^i$ is to choose move "a" at every player i position where p holds. At positions where p does not hold, the strategy is allowed to choose any enabled move. $\sigma_1 + \sigma_2$ says that the strategy of player i conforms to the specification σ_1 or σ_2. The construct $\sigma_1 \cdot \sigma_2$ says that the strategy conforms to specifications σ_1 and σ_2.

The specification $\pi \Rightarrow \sigma$ says, at any node player i sticks to the specification given by σ if on the history of the play, all moves made by $\bar{\imath}$ conform to π. Here again, which player plays π and which one plays σ is determined by the atomic formulas used in the specifications. In strategies, this captures the aspect of players' actions being responses to the opponent's moves. The opponent's complete strategy may not be available, the player makes a choice taking into account the apparent behaviour of the opponent on the history of play.

Let $\Sigma = \{a_1, \ldots, a_m\}$ be the set of all moves, we also make use of the following abbreviation.

- $\mathit{null}^i = [\top \mapsto a_1]^i + \cdots + [\top \mapsto a_m]^i$.

where $\top = p \vee \neg p$ for an observable $p \in P^i$. It will be clear from the semantics that any strategy of player i conforms to null^i, or in other words this is an empty specification. The empty specification is particularly useful for assertions of the form "there exists a strategy" where the property of the strategy is not of any relevance.

Semantics Given any sequence $\xi = t_0 t_1 \cdots t_m$, $V : \{t_0, \cdots, t_m\} \to 2^X$, and k such that $0 \leq k \leq m$, the truth of a past formula $\psi \in \mathit{Past}(X)$ at k, denoted $\xi, k \models \psi$ is defined as follows, as is standard in tense logic:

- $\xi, k \models p$ iff $p \in V(t_k)$.
- $\xi, k \models \neg\psi$ iff $\xi, k \not\models \psi$.
- $\xi, k \models \psi_1 \vee \psi_2$ iff $\xi, k \models \psi_1$ or $\xi, k \models \psi_2$.
- $\xi, k \models \diamondminus\psi$ iff there exists a $j : 0 \leq j \leq k$ such that $\xi, j \models \psi$.

We consider the game arena $\mathcal{G}$ along with a valuation function for the observables $V : W \to 2^P$. Given a strategy μ of player i and a node $s \in \mu$, let $\rho_s : s_0 a_0 s_1 \cdots s_m = s$ be the unique path in μ from the root node to s. For a strategy specification $\sigma \in \mathit{Strat}^i(P^i)$, we define when μ conforms to σ (denoted $\mu \models_i \sigma$) as follows:

- $\mu \models_i \sigma$ iff for all player i nodes $s \in \mu$, we have $\rho_s, s \models_i \sigma$.

where we define $\rho_s, s_j \models_i \sigma$ for any player i node s_j in ρ_s as,

- $\rho_s, s_j \models_i [\psi \mapsto a]^i$ iff $\rho_s, j \models \psi$ implies $\mathit{out}_{\rho_s}(s_j) = a$.
- $\rho_s, s_j \models_i \sigma_1 + \sigma_2$ iff $\rho_s, s_j \models_i \sigma_1$ or $\rho_s, s_j \models_i \sigma_2$.
- $\rho_s, s_j \models_i \sigma_1 \cdot \sigma_2$ iff $\rho_s, s_j \models_i \sigma_1$ and $\rho_s, s_j \models_i \sigma_2$.
- $\rho_s, s_j \models_i \pi \Rightarrow \sigma_1$ iff for all player $\bar{\imath}$ nodes $s_k \in \rho_s$ such that $k \leq j$, if $\rho_s, s_k \models_{\bar{\imath}} \pi$ then $\rho_s, s_j \models_i \sigma_1$.

Above, $\pi \in \mathit{Strat}^{\bar{\imath}}(P^1 \cap P^2)$, $\psi \in \mathit{Past}(P^i)$, and for all $i : 0 \leq i < m$, $\mathit{out}_{\rho_s}(s_i) = a_i$ and $\mathit{out}_{\rho_s}(s)$ is the unique move in μ at s.

Remarks Note that we do not have negation in specifications. One reason is that they are partial, and hence the semantics is not immediate. If we were to consider a specification of the form $\overline{\pi} \Rightarrow \sigma$, we could interpret this as: if the player has seen that the opponent has violated π in the past, then play σ. This seems rather unnatural, and hence, for the present, we are content to leave negation aside. Note that we do have negation in tests in atomic specifications, and later we will embed these specifications into a modal logic (with negation on formulas).

When we consider repeated or multi-stage games, we have strategy *switching*, whereby players receive payoffs at specified points, and depending on the outcomes, decide on what new strategies to adopt later. Then it makes sense to include specifications whereby a player conforms to a strategy until some observable change, and then switches to another strategy. In this context, we have (a form of) sequential composition as well as iteration. However, operators are best added after a systematic study of their algebraic properties. We stick to a simple presentation here since our main aim is only to describe the framework. As we will see below, any set of specifications that allows effective automaton consruction will do.

Clearly, each strategy specification defines a set of strategies. We now show that it is a regular set, recognizable by a finite state device. In the spirit of prescriptive game theory, we call them advice automata.

Advice Automata For a game arena $\mathcal{G}$, a nondeterministic advice automaton for player i is a tuple $\mathcal{A} = (Q, \delta, o, I)$ where Q is the set of states, $I \subseteq Q$ is the set of initial states, $\delta : Q \times W \times \Sigma \rightarrow 2^Q$ is the transition relation, and $o : Q \times W^i \rightarrow \Sigma$, is the output or advice function.

The language accepted by the automaton is a set of strategies of player i. Given a strategy $\mu = (W_\mu, \longrightarrow_\mu, s_0)$ of player i, a run of $\mathcal{A}$ on μ is a Q labelled tree $T = (W_\mu, \longrightarrow_\mu, \lambda)$, where λ maps each tree node to a state in Q as follows: $\lambda(s_0) \in I$, and for any s_k where $s_k \xrightarrow{a_k}_\mu s'_k$, we have $\lambda(s'_k) \in \delta(\lambda(s_k), s_k, a_k)$.

A Q-labelled tree T is accepted by $\mathcal{A}$ if for every tree node $s \in W^i_\mu$, if $s \xrightarrow{a}_T s'$ then $o(\lambda(s), s) = a$. A strategy μ is accepted by $\mathcal{A}$ if there exists an accepting run of $\mathcal{A}$ on μ.

It is easy to see that any bounded memory strategy can be represented using a deterministic advice automaton. In such a framework we can ask, given a bounded memory strategy for player 2 represented by a deterministic strategy automaton B, can we compute the best response for player 1?

Proposition 3.1. *Given a game $G = (\mathcal{G}, \mathcal{E})$ and a deterministic advice automaton B for player 2, the best response for player 1 can be effectively computed (in the form of an advice automaton).*

The proposition is proved easily. For each $F \in 2^R$ (cf. page 10), we can construct a nondeterministic automaton A_F which explores paths of $\mathcal{G}$ as follows. It consults B to pick player 2's moves and simply guesses 1's moves. It runs the binary evaluation automaton $\mathcal{E}^1_F$ for player 1 in parallel and checks if the run is winning for player 1. We can enumerate $F \in 2^R$ in such a way that those higher in $\lhd^1$ appear earlier in the enumeration. We try automata A_F in this order.

Therefore, given a strategy profile presented as advice automaton for each of the players, we can also check if a strategy profile constitutes a Nash equilibrium. However, we are interested in strategy specifications which are partial and hence constitute nondeterministic advice automata. The following lemma relates structured strategy specifications to advice automata.

Lemma 3.1. *Given a player $i \in \{1, 2\}$ and a strategy specification σ, we can construct an advice automaton $\mathcal{A}_\sigma$ such that $\mu \in Lang(\mathcal{A}_\sigma)$ iff $\mu \models_i \sigma$.*

Proof. The construction of automata is inductive, on the structure of specifications. Note that the strategy is implemented principally by the output function of the advice automaton.

For a strategy specification σ, let $SF(\sigma)$ denote the subformula closure of σ and $SF_\psi(\sigma)$ denote the *Past* subformulas in σ. Call $R \subseteq SF_\psi(\sigma)$ an atom if it is propositionally consistent and complete: that is, for every $\neg\gamma \in SF_\psi(\sigma)$, $\neg\gamma \in R$ iff $\gamma \notin R$, and for every $\gamma_1 \vee \gamma_2 \in SF_\psi(\sigma)$, $\gamma_1 \vee \gamma_2 \in R$ iff $\gamma_1 \in R$ or $\gamma_2 \in R$.

Let $\mathcal{AT}_\sigma$ denote the set of atoms. Let $C_0 = \{C \in \mathcal{AT}_\sigma |$ there does not exist any $\diamondminus\psi \in C\}$. For $C, D \in \mathcal{AT}_\sigma$, define $C \longrightarrow D$ iff for all $\diamondminus\psi \in SF_\psi(\sigma)$, the following conditions hold.

– $\psi \in C \Rightarrow \diamondminus\psi \in D$
– $\diamondminus\psi \in D \Rightarrow \psi \in C$ or $\diamondminus\psi \in C$.

We proceed by induction on the structure of σ. We construct automata for atomic strategies and compose them for complex strategies.
($\sigma \equiv [\psi \mapsto a]$): The automaton works as follows. Its states keep track of past formulas satisfied along a play as game positions are traversed and that the valuation respects the constraints generated for satisfying ψ. The automaton also guesses a move at every step and checks that this is indeed a when ψ holds; in such a case this is the output of the automaton. Formally:
$\mathcal{A}_\sigma = (Q_\sigma, \delta_\sigma, o_\sigma, I_\sigma)$, where

– $Q_\sigma = \mathcal{AT}_\sigma \times \Sigma$.
– $I_\sigma = \{(C, x) | C \in C_0, V(s_0) = C \cap P_\sigma, x \in \Sigma\}$.
– For a transition $s \xrightarrow{a} s'$ in $\mathcal{G}$, we have:
$\delta_\sigma((C, x), s, a) = \{(C', y) | C \longrightarrow C', V(s') = C' \cap P_\sigma, y \in \Sigma\}$.
– $o((C, x), s) = \begin{cases} a \text{ if } \ \psi \in C \\ x \text{ otherwise} \end{cases}$

We now prove the assertion in the lemma that $\mu \in Lang(\mathcal{A}_\sigma)$ iff $\mu \models_i \sigma$.

($\Rightarrow$) Suppose $\mu \in Lang(\mathcal{A}_\sigma)$. Let $T = (W^1_\mu, W^2_\mu, \longrightarrow_T, \lambda)$ be the Q-labelled tree accepted by $\mathcal{A}_\sigma$. We need to show that for all $s \in W_\mu$, we have $\rho_s, s \models \psi$ implies $out(s) = a$.

The following claim, easily proved by structural induction on the structure of ψ, using the definition of $\longrightarrow$ on atoms, asserts that the states of the automaton check the past requirements correctly. Below we use the notation $\psi \in (C, x)$ to mean $\psi \in C$.

Claim. For all $s \in W_\mu$, for all $\psi' \in SF_\psi(\sigma)$, $\psi' \in \lambda(s)$ iff $\rho_s, s \models \psi'$.

Assume the claim and consider any $s \in W_\mu$. We have $\rho_s, s \models \psi$ implies $\psi \in \lambda(s)$. By the definition of o, we have $o(\lambda(s), s) = a$.

($\Leftarrow$) Suppose $\mu \models_1 [\psi \mapsto a]$. From the semantics, we have $\forall s \in W^1_\mu, \rho_s, s \models \psi$ implies $out(s) = a$. We need to show that there exists a Q-labelled tree accepted by $\mathcal{A}_\sigma$. For any s let the Q-labelling be defined as follows. Fix $x_0 \in \Sigma$.

– For $s \in W^1_\mu$, let $\lambda(s) = (\{\psi' \in SF_\psi(\sigma) | \rho_s, s \models \psi'\}, out(s))$.

– For $s \in W^2_\mu$, let $\lambda(s) = (\{\psi' \in SF_\psi(\sigma) | \rho_s, s \models \psi'\}, x_0)$.

It is easy to check that $\lambda(s)$ constitutes an atom and the transition relation is respected. By the definition of o, we get that it is accepting.
($\sigma \equiv \sigma_1 \cdot \sigma_2$): By induction hypothesis there exist $\mathcal{A}_{\sigma_1} = (Q_{\sigma_1}, \delta_{\sigma_1}, o_{\sigma_1}, I_{\sigma_1})$ and $\mathcal{A}_{\sigma_2} = (Q_{\sigma_2}, \delta_{\sigma_2}, o_{\sigma_2}, I_{\sigma_2})$ which accept all strategies satisfying σ_1 and σ_2 respectively. To obtain an automaton which accepts all strategies which satisfy $\sigma_1 \cdot \sigma_2$ we just need to take the product of $\mathcal{A}_{\sigma_1}$ and $\mathcal{A}_{\sigma_2}$.
($\sigma \equiv \sigma_1 + \sigma_2$): We take $\mathcal{A}_\sigma$ to be the disjoint union of $\mathcal{A}_{\sigma_1}$ and $\mathcal{A}_{\sigma_2}$. Since the automaton is nondeterministic with multiple initial states, we retain the intial states of both $\mathcal{A}_{\sigma_1}$ and $\mathcal{A}_{\sigma_2}$. If a run starts in an initial state of $\mathcal{A}_{\sigma_1}$ then it will never cross over into the state space of $\mathcal{A}_{\sigma_2}$ and vice versa.
($\sigma \equiv \pi \Rightarrow \sigma'$): By induction hypothesis we have $\mathcal{A}_\pi = (Q_\pi, \delta_\pi, o_\pi, I_\pi)$ which accepts all player 2 strategies satisfying π and $\mathcal{A}_{\sigma'} = (Q_{\sigma'}, \delta_{\sigma'}, o_{\sigma'}, I_{\sigma'})$ which accepts all player 1 strategies satisfying σ'.

The automaton $\mathcal{A}_\sigma$ has the product states of $\mathcal{A}_\pi$ and $\mathcal{A}_{\sigma'}$ as its states along with a special state q_{free}. The automaton keeps simulating both $\mathcal{A}_\pi$, $\mathcal{A}_{\sigma'}$ and keeps checking if the path violates the advice given by $\mathcal{A}_\pi$, if so it moves into state q_{free} from which point onwards it is “free” to produce any advice. Till π is violated, it is forced to follow the transitions of $\mathcal{A}_{\sigma'}$.

Define $\mathcal{A}_\sigma = (Q, \delta, o, I)$ where $Q = (Q_\pi \times Q_{\sigma'}) \cup (q_{\mathit{free}} \times \Sigma)$. The transition function is given as follows:

– For $s \in W^1_\mu$, we have $\delta((q_\pi, q_{\sigma'}), s, a) = \{(q_1, q_2) | q_1 \in \delta_\pi(q_\pi, s, a)$ and $q_2 \in \delta_{\sigma'}(q_{\sigma'}, s, a)\}$.
– For $s \in W^2_\mu$, we have:
  - If $o_\pi(q_\pi, s) \neq a$, then $\delta((q_\pi, q_{\sigma'}), s, a) = \{(q_{\mathit{free}}, a) | a \in \Sigma\}$.
  - If $o_\pi(q_\pi, s) = a$, then $\delta((q_\pi, q_{\sigma'}), s, a) = \{(q_1, q_2) | q_1 \in \delta_\pi(q_\pi, s, a)$ and $q_2 \in \delta_{\sigma'}(q_{\sigma'}, s, a)\}$.
– $\delta((q_{\mathit{free}}, x), s, a) = \{(q_{\mathit{free}}, a) | a \in \Sigma\}$

The output function is defined as follows: For $s \in W^1_\mu$, $o((q_\pi, q_{\sigma'}), s) = o_{\sigma'}(q_{\sigma'}, s)$ and $o((q_{\mathit{free}}, x), s) = x$.

The automaton keeps simulating both $\mathcal{A}_\pi$, $\mathcal{A}_{\sigma'}$ and keeps checking if the path violates π. If so it moves into state q_{free} from which point onwards it is not constrained to follow σ'.

3.3 Best response

Since a strategy specification denotes a set of strategies satisfying certain properties, notions like strategy comparison and best response with respect to strategy specifications need to be redefined.

Given a game $G = (\mathcal{G}, \mathcal{E})$ and a strategy specification π for player $\bar{i}$, we can have different notions as to when a specification for player i is "better" than another.

- $\mathit{Better}_1(\sigma, \sigma')$: if $\exists F \in 2^R, \exists \mu'$ with $\mu' \models_i \sigma'$ such that $\forall \tau$ with $\tau \models_{\bar{i}} \pi$, $\rho^{\tau}_{\mu'}$ is winning with respect to $\mathcal{E}^i_F$ then $\exists \mu$ with $\mu \models_i \sigma$ such that $\forall \tau$ with $\tau \models_{\bar{i}} \pi$, ρ^{τ}_{μ} is winning with respect to $\mathcal{E}^i_F$.
  The predicate $\mathit{Better}_1(\sigma, \sigma')$ says that, for some (binary) outcome F, if there is a strategy conforming to the specification σ' which ensures winning $\mathcal{E}^i_F$ then there also exists a strategy conforming to σ which ensures winning $\mathcal{E}^i_F$ as well.
- $\mathit{Better}_2(\sigma, \sigma')$: if $\exists F \in 2^R$ such that $\forall \mu'$ with $\mu' \models_i \sigma'$, $\forall \tau$ with $\tau \models_{\bar{i}} \pi$, $\rho^{\tau}_{\mu'}$ is winning with respect to $\mathcal{E}^i_F$ then $\forall \mu$ with $\mu \models_i \sigma$, $\forall \tau$ with $\tau \models_{\bar{i}} \pi$, ρ^{τ}_{μ} is winning with respect to $\mathcal{E}^i_F$.
  This notion is best understood contrapositively: for some (binary) outcome F, whenever there is a strategy conforming to σ which is not winning for $\mathcal{E}^i_F$, there also exists a strategy conforming to σ' which is not winning for $\mathcal{E}^i_F$. This can be thought of as a soundness condition. A risk averse player might prefer this way of comparison.

To algorithmically compare strategies, we first need to be able to decide the following questions. Let σ and π be strategy specifications for player i and player $\bar{i}$ and $\mathcal{E}^i_F$ a binary evaluation automaton for player i.

- Does player i have a strategy conforming to σ which ensures a valid play which is winning for i with respect to $\mathcal{E}^i_F$, as long as player $\bar{i}$ is playing a strategy conforming to π (abbreviated as $\exists\sigma, \forall\pi : \mathcal{E}^i_F$)?
- Is it the case that for all strategies of player i conforming to σ, as long as player $\bar{i}$ is playing a strategy conforming to π, the result will be a valid play which is winning for i with respect to $\mathcal{E}^i_F$ (abbreviated as $\forall\sigma, \forall\pi : \mathcal{E}^i_F$)?

We call this the *verification* question. The *synthesis* question is given π and $\mathcal{E}^i_F$ to construct a specification σ such that $\exists\sigma, \forall\pi : \mathcal{E}^i_F$ holds.

Once we can show that the verification question is decidable and synthesis possible, the game theoretic questions of interest include: For a game $G = (\mathcal{G}, \mathcal{E})$,

- Given strategy specifications σ and π, check if σ is a best response to π.
- Given a strategy specification profile $\langle \sigma, \pi \rangle$, check if it is a Nash equilibrium.
- Given a strategy specification π for player $\bar{i}$ and $F \in F$, synthesize (if possible) a specification σ for i such that $\exists\sigma, \forall\pi : \mathcal{E}^i_F$ holds.
- Given a strategy specification π for $\bar{i}$, synthesize a specification σ such that σ is the best response to π.

The main theorem of the section is the following assertion.

Theorem 3.1. *Given a game $G = (\mathcal{G}, \mathcal{E})$ and a strategy specification π for player $\bar{i}$,*

1. *The verification problem of checking whether for a player i strategy specification σ and a binary evaluation automaton $\mathcal{E}_F^i$, if $\exists\sigma, \forall\pi : \mathcal{E}_F^i$ and $\forall\sigma, \forall\pi : \mathcal{E}_F^i$ hold in $\mathcal{G}$ is decidable.*
2. *For a binary evaluation automaton $\mathcal{E}_F^i$, it is possible to synthesize (when one exists) a deterministic advice automaton $\mathcal{A}_i$ such that $\mathcal{A}_i, \forall\pi : \mathcal{E}_F^i$ holds.*
3. *For a specification σ, checking if σ is the best response to π is decidable.*
4. *It is possible to synthesize a deterministic advice automaton $\mathcal{A}_i$ such that $\mathcal{A}_i$ is the best response to π.*

For an advice automaton $\mathcal{A}_i$, we can define the restriction of $\mathcal{G}$ with respect to $\mathcal{A}_i$ by removing all nodes and edges that are not reachable when play proceeds according to $\mathcal{A}_i$. Thus we can also define $\mathcal{G} \upharpoonright \mathcal{A}_\pi$, for a strategy specification π. The restricted arena is no longer deterministic. However, for any player 2 node in $\mathcal{G} \upharpoonright \mathcal{A}_\pi$ there is exactly one action enabled.

(1): To check if $\exists\sigma, \forall\pi : \mathcal{E}_F^i$ holds, we build a non-deterministic tree automaton $\mathcal{T}$ which runs on $\mathcal{G} \upharpoonright \mathcal{A}_\pi$. For a 1 node, it guesses an action "a" which conforms to σ and branches out on all a edges. For a 2 node, there is only one action enabled in $\mathcal{G} \upharpoonright \mathcal{A}_\pi$, call the action b. The automaton branches out on all b labelled edges. $\mathcal{T}$ runs $\mathcal{E}_F^1$ in parallel to verify that all plays thus constructed are winning for 1 with respect to $\mathcal{E}_F^1$. If $\mathcal{T}$ has an accepting run, then $\exists\sigma, \forall\pi : \mathcal{E}_F^i$ holds in $\mathcal{G}$.

(2): We want a deterministic advice automaton $\mathcal{A}_1$ which ensures that for all strategies of 2 conforming to π the play is "winning" for player 1. We construct a tree automaton $\mathcal{T}$ which mimics the subset construction to synthesize $\mathcal{A}_1$. The states of $\mathcal{T}$ are the subsets of states of $\mathcal{A}_\pi$. At a game position of player 1, it guesses a move and for every player 2 game position, it branches out on all the action choices of $\mathcal{A}_\pi$ where for each move the resulting new state is the subset of states given by the nondeterministic transition relation of $\mathcal{A}_\pi$. $\mathcal{T}$ runs $\mathcal{E}_F^1$ in parallel and checks if all paths constitute a valid play and that the play is winning for 1 with respect to $\mathcal{E}_F^1$. If there is an accepting run for $\mathcal{T}$, then constructing $\mathcal{A}_1$ is easy. The state space of $\mathcal{A}_1$ is the set of all subsets of the states of $\mathcal{A}_\pi$. The transition relation is derived from the usual subset construction performed by $\mathcal{T}$. The output function basically follows the accepting run of $\mathcal{T}$.

(3): Given σ and π to check if σ is the best response to π, we use the tree automaton construction in (1) with a slight modification.

We enumerate the elements of 2^R in such a way that those higher in $\lhd^1$ appear earlier in the enumeration. For each F, we construct a tree automaton as in (1), the only difference being that the guesses made by $\mathcal{T}$ at player 1 game positions are not restricted by σ. $\mathcal{T}$ runs $\mathcal{E}_F^1$ in parallel to check if player 1 can ensure F for all choices of 2 which conform to π. Since the evaluation automaton is "complete", the play eventually settles down in one of $F' \in 2^R$. Therefore, as we try elements of 2^R in order, the tree automaton succeeds for some $\mathcal{E}_{F'}^1$.

This gives us the "best" outcome which player 1 can guarantee. We then check if $\exists\sigma, \forall\pi : \mathcal{E}^1_{F'}$ holds in $\mathcal{G}$. If it does then $\mathcal{A}_\sigma$ is a best response to $\mathcal{A}_\pi$.

This also implies that we can check whether a strategy profile (presented as advice automata) constitutes a Nash equilibrium.

(4) is similar to (3). We enumerate 2^R and find the "best" outcome that can be achieved and using the synthesis procedure, synthesize an advice automaton for this outcome.

This proof sketch hopefully gives the reader some idea of how automata theory is employed gainfully in strategy synthesis. Note the essential use of memory structure in strategies. For details, the reader is referred to [39].

4 Game logic and its descendants

In Section 2 we have argued that resource bounded players strategize locally using heuristic methods. This calls for a study of the compositional structure of strategies where logic provides a useful tool. We now move on to the logical studies of compositional games and strategies. In this section we discuss composite game structures, where strategies are embedded in the models of the proposed logics. These strategies take up an existential role in giving meaning to the game operators in the language. Studies on modeling strategies explicitly in the logical language will be taken up in the next section.

To look at compositional structure in games, viewing games as programs becomes useful. We first give a brief introduction to a logic of programs, based on which different game logics were proposed.

4.1 Propositional Dynamic Logic

As mentioned in [18], we can define a computer program as follows: *a recipe written in a formal language for computing desired output data from given input data.* Propositional Dynamic Logic (*PDL*) is a logic of programs (non-deterministic) where programs are made explicit in the language. Complex programs are built out of basic programs using some binary program constructs like $\cup$ (choice) and ; (sequential composition) and unary construct $*$ (iteration). For a detailed introduction to *PDL*, see [18,4]. The language of *PDL* is given as follows:

Definition 4.1. *Given a set of atomic program terms Π and a set of atomic propositions Φ, program terms π and formulas φ are defined inductively:*

$$\pi := b \mid \pi;\pi \mid \pi \cup \pi \mid \pi^*$$
$$\varphi := \bot \mid p \mid \neg\varphi \mid \varphi \vee \varphi \mid [\pi]\varphi,$$

where $p \in \Phi$, and $b \in \Pi$.

If π_1 and π_2 are programs, the program $\pi_1 \cup \pi_2$ nondeterministically executes π_1 or π_2, $\pi_1;\pi_2$ is a program that first executes π_1 and then π_2, π^* is a program that execute π a finite (possibly zero) number of times.

A model for the language of *PDL*, viz. a program model is of the form $\mathcal{M} = \langle S, \{\mathcal{R}_\pi : \pi \text{ is a program}\}, V\rangle$, where S is a non-empty set of states, each $\mathcal{R}_\pi$ is a binary relation over S, and V is a valuation assigning truth values to atomic propositions in states. Alternatively, one can also think of $\mathcal{R}_\pi$'s as maps from S to 2^S. Let us now suppose that the relations corresponding to the composite program constructs are constructed as follows:

$$\mathcal{R}_{\pi_1 \cup \pi_2} := \mathcal{R}_{\pi_1} \cup \mathcal{R}_{\pi_2}$$
$$\mathcal{R}_{\pi_1;\pi_2} := \mathcal{R}_{\pi_1} \circ \mathcal{R}_{\pi_2} \ (= \{(x,y) : \exists z(\mathcal{R}_{\pi_1}xz \text{ and } \mathcal{R}_{\pi_2}zy)\})$$
$$\mathcal{R}_{\pi^*} := (\mathcal{R}_\pi)^*, \text{ the reflexive transitive closure of } \mathcal{R}_\pi$$

Truth of a formula φ in a model $\mathcal{M}$ at a state s is defined as follows:

$\mathcal{M}, s \models p$ iff $s \in V(p)$
$\mathcal{M}, s \not\models \bot$
$\mathcal{M}, s \models \neg\varphi$ iff $\mathcal{M}, s \not\models \varphi$
$\mathcal{M}, s \models \varphi \vee \psi$ iff $\mathcal{M}, s \models \varphi$ or $\mathcal{M}, s \models \psi$
$\mathcal{M}, s \models [\pi]\varphi$ iff for all s': $\mathcal{R}_\pi ss'$, $\mathcal{M}, s' \models \varphi$

As given earlier, Π denotes the set of basic programs. Let $\widehat{\Pi}$ be the smallest set containing Π and closed under the program constructs $\cup$, ; and $*$. This is the class of regular or finite-state programs. Kleene [22] showed that these operations suffice to capture all finite-state behaviours.

In the following we are not distinguishing between a program π and its interpretation $\mathcal{R}_\pi$. As mentioned earlier, a program π can be thought of as a map from S to 2^S, $\pi(s)$ denoting the set of states that the program may reach, when started operation from s. Alternatively, if we were given a set of states X, we can consider the program to be a mechanism that 'achieves' X starting from s. If we consider X as a goal, a program then sounds like a strategy to achieve the goal, and programs can be thought of as 1-player games.

Consider the program $(a+b); (e+f)$. This refers to a player (say Nature) nondeterministically choosing between actions a or b, followed by choosing actions e or f. But, if we consider a second player in our discussion, the same composite structure $(a + b); (e + f)$, could be considered as a game between two players I and II, where player I chooses to do either an a or b, and then player II chooses to do e or f . One can then think of it as a sequential composition of two one player games $(a + b)$ and $(e + f)$ with rôles of the player and the opponent 'switched' in the two games. This idea leads us to a propositional game logic (cf. Section 4.2), which is similar to program logic, but admitting a player and an opponent.

4.2 Game Logic

Game Logic (*GL*), which was proposed in [28] studies how a player's 'power' evolves in a two-player game. We talk about two person zero sum games of perfect information in this logic. Similar to the language of *PDL*, the language of *GL* is defined as follows:

Definition 4.2. *Given a set of atomic games Γ and a set of atomic propositions Φ, game terms γ and formulas φ are defined inductively:*

$\gamma := g \mid \varphi? \mid \gamma;\gamma \mid \gamma \cup \gamma \mid \gamma^d \mid \gamma^*$
$\varphi := \bot \mid p \mid \neg\varphi \mid \varphi \vee \varphi \mid \langle\gamma\rangle\varphi,$

where $p \in \Phi$, and $g \in \Gamma$.

Here we consider a set of atomic games Γ, and the following constructs which form new games: choice $(\gamma \cup \gamma')$, dual (γ^d), sequential composition $(\gamma; \gamma')$, and iteration (γ^*). The game γ^d is obtained when the game γ is played with the players switching rôles.

The intuitive reading of the formula $\langle\gamma\rangle\varphi$ is 'player 1 has a strategy in game γ to ensure φ'. Here we only consider the final outcomes which players can enforce in the games. This is modeled by the notion of effectivity relations between states and sets of states. An effectivity relation E_g, corresponding to an atomic game g, on a state space S is a subset of $S \times 2^S$ whose intuitive reading is:

> $(s, X) \in E_g$ iff starting at s, in game g, player 1 can enforce the outcome to be in the set X.

Note that it is enough to assign effectivity relation to one of the players, as for the opponent player the corresponding relation is given by the dual game. In other words, these effectivity relations satisfy the condition of determinacy: If it is not the case that player 1 can enforce the outcome to be in the set X, then player 2 can enforce the outcome to be in the set X^c, the complement of X and vice versa.

Definition 4.3. *A game model is a structure $\mathcal{M} = (S, \{E_g \mid g \in \Gamma\}, V)$, where S is a set of states, V is a valuation assigning truth values to atomic propositions in states, and for each $g \in \Gamma$, $E_g \subseteq S \times 2^S$. We assume that for each g, the relations are upward closed under supersets (Monotonicity condition).*

The truth definition for formulas φ in a model $\mathcal{M}$ at a state s is standard, except for the modality $\langle\gamma\rangle\varphi$, and they are given as follows:

$\mathcal{M}, s \models p$ iff $s \in V(p)$
$\mathcal{M}, s \not\models \bot$
$\mathcal{M}, s \models \neg\varphi$ iff $\mathcal{M}, s \not\models \varphi$
$\mathcal{M}, s \models \varphi \vee \psi$ iff $\mathcal{M}, s \models \varphi$ or $\mathcal{M}, s \models \psi$
$\mathcal{M}, s \models \langle\gamma\rangle\varphi$ iff there exists $X \subseteq S : sE_\gamma X$ and for all $x \in X : \mathcal{M}, x \models \varphi$.

The semantics is standard and is generally termed in the existing literature as neighborhood models [8]. Suppose $E_\gamma(X) = \{s \in S \mid sE_\gamma X\}$. The effectivity conditions for players in complex two-person games are as follows:

$E_{\gamma\cup\gamma'}(X) = E_\gamma(X) \cup E_{\gamma'}(X)$
$E_{\gamma^d}(X) = S \setminus E_\gamma(S \setminus X)$
$E_{\gamma;\gamma'}(X) = E_\gamma(E_{\gamma'}(X))$
$E_{\gamma^*}(X) = \mu Y.X \cup E_\gamma(Y)$

Monotonicity of E_g's is preserved under game operations and hence the fixpoint $\mu Y.X \cup E_\gamma(Y)$ always exists. A formula φ is satisfiable if there exists a model $\mathcal{M}$ and a state s such that $\mathcal{M}, s \models \varphi$. A formula φ is valid if it is true in every model.

We should note here that in GL the two players cannot have winning strategies for complementary winning positions, thus $\neg(\langle\gamma\rangle\varphi \wedge \langle\gamma^d\rangle\neg\varphi)$ is a valid formula, for every game γ. All games are determined, that is, in any game, one of the players has a winning strategy . Thus, $(\langle\gamma\rangle\varphi \vee \langle\gamma^d\rangle\neg\varphi)$ is also a valid formula in this logic. The following gives an axiom system for this logic.

Axiom system

(a) all propositional tautologies and inference rules

(b) reduction axioms:
$\langle \gamma \cup \gamma' \rangle \varphi \leftrightarrow \langle \gamma \rangle \varphi \vee \langle \gamma' \rangle \varphi$
$\langle \gamma^d \rangle \varphi \leftrightarrow \neg \langle \gamma \rangle \neg \varphi$
$\langle \gamma ; \gamma' \rangle \varphi \leftrightarrow \langle \gamma \rangle \langle \gamma' \rangle \varphi$
$\langle \gamma^* \rangle \varphi \leftrightarrow \varphi \vee \langle \gamma \rangle \langle \gamma^* \rangle \varphi$

Inference rules

$$(MP)\ \frac{\varphi, \quad \varphi \to \psi}{\psi} \qquad (NG)\ \frac{\varphi \to \psi}{\langle \gamma \rangle \varphi \to \langle \gamma \rangle \psi}$$

$$(IND)\ \frac{\langle \gamma \rangle \varphi \to \varphi}{\langle \gamma^* \rangle \varphi \to \varphi}$$

Completeness and Decidability The soundness of the axiom system can be proved easily. The system without the duality axiom can be proved to be complete for the dual-free fragment of the logic [28]. The system without the iteration axiom and rule can be proved to be complete for the iteration-free fragment of the logic [34]. In [28], Parikh conjectured that the system presented is indeed complete for game logic. This remains an interesting open problem.

The satisfiability problem for the logic above is EXPTIME-complete. This is the same as that for *PDL*. Model checking game logic is equivalent to the same problem for the modal μ-calculus. Complexity of model checking is in NP $\cap$ co-NP. The details of these results can be found in [34]. A major open problem asks if the complexity of model checking is in P.

4.3 Parallel composition: intersecting

In this section we study a parallel composition operator of two player games, the underlying idea of which is to consider players' powers while playing simultaneous games. In game theory, typical matrix games like 'Prisoner's Dilemma' involve simultaneous moves for two players: each chooses independently from the other, and the outcome may be viewed as the set of both moves. In another setting, computer scientist use parallel games with simultaneous moves to model concurrent processes.

Here, we will model simultaneous play of parallel games in terms of players' abstract powers, without considering communication. Using ideas from propositional dynamic logic for concurrency [16], a system is proposed where players' powers in a parallel game can be reduced to their powers in the constituent games. The details of this study can be found in [2].

Effectivity relations for product games While considering simultaneous games, we first note that games can produce complex outcome states, denoted by sets read ‘conjunctively’ as in Concurrent Propositional Dynamic Logic (*CPDL*), developed in [16]. But in addition, players can also have choices leading to sets of such sets, still read disjunctively at this second level as one does in *GL* (see the intuitive reading of E_g in Section 4.2). With this idea, to have an intuitive model for simultaneous games, we consider effectivity relations E_g for atomic games g to be subsets of $S \times 2^{2^S}$. Suppose X, U, T, W range over sets of sets of states, t, w range over sets of states, and s, u range over states:

Effectivity relations, E^i (for player i) for composite games are given as follows:

$$\begin{array}{rlc}
sE^1_{\gamma\cup\gamma'}X & \text{iff} & sE^1_{\gamma}X \text{ or } sE^1_{\gamma'}X \\
sE^2_{\gamma\cup\gamma'}X & \text{iff} & sE^2_{\gamma}X \text{ and } sE^2_{\gamma'}X \\
sE^1_{\gamma^d}X & \text{iff} & sE^2_{\gamma}X \\
sE^2_{\gamma^d}X & \text{iff} & sE^1_{\gamma}X \\
sE^i_{\gamma;\gamma'}X & \text{iff} & \exists U : sE^i_{\gamma}U \text{ and for each } u \in \bigcup U,\ uE^i_{\gamma'}X \\
sE^i_{\gamma\times\gamma'}X & \text{iff} & \exists T,\ \exists W : sE^i_{\gamma}T \text{ and } sE^i_{\gamma'}W \\
 & & \text{and } X = \{t \cup w : t \in T \text{ and } w \in W\}
\end{array}$$

As an illustration, we show how this format for computation of players’ powers fits an intuitive example of parallel games, for instance, simultaneous move selection in a matrix game:

$$\begin{tikzcd}
 & 1 \arrow[dl] \arrow[dr] & & & 2 \arrow[dl] \arrow[dr] & \\
a & \mathbf{G} & b & c & \mathbf{H} & d
\end{tikzcd}$$

To make things comparable, we now change earlier single outcomes s to singleton states s. The powers of 1 in the game **G** are given by {{a}}, {{b}} and that of 2 by {{a}, {b}}. Similarly, in the game **H**, the powers of 1 and 2 are {{c}, {d}} and {{c}}, {{d}}, respectively. The powers of 1 and 2 in the product game $\mathbf{G} \times \mathbf{H}$ are then formed by taking unions: {{a, c}, {a, d}}, {{b, c}, {b, d}} and {{a, c}, {b, c}}, {{a, d}, {b, d}}, respectively. Reading the inner brackets as conjunctive, and the outer ones as disjunctive, this seems to fit our intuitions.

We should note here that we now have separate effectivity relations for each player. This is a result of considering simultaneous games where the determinacy condition of *GL* (cf. Section 4.2) fails.

Concurrent Dynamic Game Logic The language of Concurrent Dynamic Game Logic (*CDGL*) is given as follows:

Definition 4.4. *Given a set of atomic games Γ and atomic propositions Φ,* game terms γ *and* formulas φ *are defined inductively as:*

$$\gamma := g \mid \gamma^d \mid \gamma;\gamma \mid \gamma \cup \gamma \mid \gamma \times \gamma$$
$$\varphi := \bot \mid p \mid \neg\varphi \mid \varphi \vee \varphi \mid \langle \gamma, i \rangle \varphi$$

where we take $p \in \Phi$, $g \in \Gamma$ and $i \in \{1, 2\}$.

For the sake of simplicity we are not considering 'iteration' here. The intended meaning of the new game construct $\gamma \times \gamma'$ is that the games γ and γ' are played in parallel, without communication. We are considering a very simple level of abstraction in describing simultaneous games, and as such not considering the complicated interaction aspects of parallel game playing.

Definition 4.5. *A conjunctive game model is a structure $\mathcal{M} = (S, \{E^i_g \mid g \in \Gamma\}, V)$, where S is a set of states, V is a valuation assigning truth values to atomic propositions in states, and with basic relations $E^i_g \subseteq S \times 2^{2^S}$ assigned to basic game expressions g, satisfying the conditions of Monotonicity, Consistency and Non-Triviality, given below:*

(C1) Monotonicity: If $sE^i_g X$ and $X \subseteq X'$, then $sE^i_g\ X'$.
(C2) Consistency: If $sE^I_g Y$ and $sE^{II}_g Z$, then Y and Z overlap.
(C3) Non-Triviality: No player can force the empty set.

These conditions can be seen as a few intuitive technical assumptions. In the semantics of the language, the truth of a formula φ in $\mathcal{M}$ at a state s is defined in the usual manner, with the following key clause for the game modality:

– $\mathcal{M}, s \models \langle \gamma, i \rangle \varphi$ iff $\exists X : sE^i_\gamma X$ and $\forall x \in \bigcup X : \mathcal{M}, x \models \varphi$.

Note that this squashes together the outcomes of all separate games, making only local assertions at single states. An alternative option would be to evaluate formulas at 'collective states', being sets of the original states. For a discussion, see [2].

Naturally, this logic encodes facts about parallel games. These are two, pointing towards an algebra of parallel games lying encoded here:

– $\langle \gamma \times \gamma', i \rangle \varphi \leftrightarrow \langle \gamma' \times \gamma, i \rangle \varphi$
– $\langle (\gamma \times \gamma')^d, i \rangle \varphi \leftrightarrow \langle \gamma^d \times \gamma'^d, i \rangle \varphi$

Axiom System

a) all propositional tautologies and inference rules
b) if $\vdash \varphi \rightarrow \psi$ then $\vdash \langle g, i \rangle \varphi \rightarrow \langle g, i \rangle \psi$
c) $\langle g, 1 \rangle \varphi \rightarrow \neg \langle g, 2 \rangle \neg \varphi$
d) $\neg \langle \gamma, i \rangle \bot$
e) reduction axioms:
$\langle \gamma \cup \gamma', 1 \rangle \varphi \leftrightarrow \langle \gamma, 1 \rangle \varphi \vee \langle \gamma', 1 \rangle \varphi$
$\langle \gamma \cup \gamma', 2 \rangle \varphi \leftrightarrow \langle \gamma, 2 \rangle \varphi \wedge \langle \gamma', 2 \rangle \varphi$
$\langle \gamma^d, 1 \rangle \varphi \leftrightarrow \langle \gamma, 2 \rangle \varphi$
$\langle \gamma^d, 2 \rangle \varphi \leftrightarrow \langle \gamma, 1 \rangle \varphi$
$\langle \gamma; \gamma', i \rangle \varphi \leftrightarrow \langle \gamma, i \rangle \langle \gamma', i \rangle \varphi$
$\langle \gamma \times \gamma', i \rangle \varphi \leftrightarrow \langle \gamma, i \rangle \varphi \wedge \langle \gamma', i \rangle \varphi$

The completeness and decidability of *CDGL* has been shown in [2]. While this result may seem to be a good advocate for *PDL*-style 'reductionism', the product axiom, $\langle \gamma \times \gamma', i\rangle\varphi \leftrightarrow \langle \gamma, i\rangle\varphi \wedge \langle \gamma', i\rangle\varphi$ also reflects the expressive poverty of *CDGL* as an account of parallelism. On one hand there are no means of stating truly collective properties of conjunctive states, on the other hand there is no scope for talking about communication or transfer of information while two or more games are played in parallel.

4.4 Parallel composition: interleaving

Consider a player playing against different opponents in two extensive form games simultaneously. Can she then have a strategy in one game using information from the other? The famous example of playing chess against two grandmasters simultaneously illustrates such reasoning. The common player in the two games acts as a conduit for transfer of information from one game to the other; thus *game composition* is essential for such reasoning. In this section (based on [14]), we consider a dynamic logic of extensive form games with sequential and parallel composition in which such situations can be expressed.

Extensive form games Let $N = \{1, \ldots, n\}$ denote the set of players, we use i to range over this set. For $i \in N$, we often use the notation $\bar{\imath}$ to denote the set $N \setminus \{i\}$. Let Σ be a finite set of action symbols representing moves of players, we let a, b range over Σ. For a set X and a finite sequence $\rho = x_1 x_2 \ldots x_m \in X^*$, let $last(\rho) = x_m$ denote the last element in this sequence. Note that in the following we will be talking about game forms (trees) only. Outcomes and players' preferences can be encoded as propositions.

Game trees Let $\mathbb{T} = (S, \Rightarrow, s_0)$ be a tree rooted at s_0 on the set of vertices S and $\Rightarrow : (S \times \Sigma) \to S$ is a partial function specifying the edges of the tree. The tree $\mathbb{T}$ is said to be finite if S is a finite set. For a node $s \in S$, let $\overrightarrow{s} = \{s' \in S \mid s \overset{a}{\Rightarrow} s'$ for some $a \in \Sigma\}$, $moves(s) = \{a \in \Sigma \mid \exists s' \in S$ with $s \overset{a}{\Rightarrow} s'\}$ and $E_T(s) = \{(s, a, s') \mid s \overset{a}{\Rightarrow} s'\}$. By $E_T(s) \times x$ we denote the set $\{((s, x), a, (s', x)) \mid (s, a, s') \in E_T(s)\}$. The set $x \times E_T(s)$ is defined similarly. A node s is called a leaf node (or terminal node) if $\overrightarrow{s} = \emptyset$. The depth of a tree is the length of the longest path in the tree.

An extensive form game tree is a pair $T = (\mathbb{T}, \widehat{\lambda})$ where $\mathbb{T} = (S, \Rightarrow, s_0)$ is a tree. The set S denotes the set of game positions with s_0 being the initial game position. The edge function $\Rightarrow$ specifies the moves enabled at a game position and the turn function $\widehat{\lambda} : S \to N$ associates each game position with a player. Technically, we need player labelling only at the non-leaf nodes. However, for the sake of uniform presentation, we do not distinguish between leaf nodes and non-leaf nodes as far as player labelling is concerned. An extensive form game tree $T = (\mathbb{T}, \widehat{\lambda})$ is said to be finite if $\mathbb{T}$ is finite. For $i \in N$, let $S^i = \{s \mid \widehat{\lambda}(s) = i\}$ and let *frontier*(T) denote the set of all leaf nodes of T. Let $S^L_T = frontier(T)$

and $S_T^{NL} = S \setminus S_T^L$. For a tree $T = (S, \Rightarrow, s_0, \widehat{\lambda})$ we use $head(T)$ to denote the depth one tree generated by taking all the outgoing edges of s_0.

A play in the game T starts by placing a token on s_0 and proceeds as follows: at any stage if the token is at a position s and $\widehat{\lambda}(s) = i$ then player i picks an action which is enabled for her at s, and the token is moved to s' where $s \overset{a}{\Rightarrow} s'$. Formally a play in T is simply a path $\rho : s_0 a_1 s_1 \cdots$ in $\mathbb{T}$ such that for all $j > 0$, $s_{j-1} \overset{a_j}{\Rightarrow} s_j$. Let $Plays(T)$ denote the set of all plays in the game tree T.

Strategies A strategy for player $i \in N$ is a function μ^i which specifies a move at every game position of the player, i.e. $\mu^i : S^i \rightarrow \Sigma$. A strategy μ^i can also be viewed as a subtree of T where for each player i node, there is a unique outgoing edge and for nodes belonging to players in $\bar{\imath}$, every enabled move is included. Formally we define the strategy tree as follows: For $i \in N$ and a player i strategy $\mu^i : S^i \rightarrow \Sigma$ the strategy tree $T_{\mu^i} = (S_{\mu^i}, \Rightarrow_{\mu^i}, s_0, \widehat{\lambda}_{\mu^i})$ associated with μ^i is the least subtree of T satisfying the following property: $s_0 \in S_{\mu^i}$,

- For any node $s \in S_{\mu^i}$,
  - if $\widehat{\lambda}(s) = i$ then there exists a unique $s' \in S_{\mu^i}$ and action a such that $s \overset{a}{\Rightarrow}_{\mu^i} s'$.
  - if $\widehat{\lambda}(s) \neq i$ then for all s' such that $s \overset{a}{\Rightarrow} s'$, we have $s \overset{a}{\Rightarrow}_{\mu^i} s'$.

Let $\Omega^i(T)$ denote the set of all strategies for player i in the extensive form game tree T. A play $\rho : s_0 a_0 s_1 \cdots$ is said to be consistent with μ^i if for all $j \geq 0$ we have $s_j \in S^i$ implies $\mu^i(s_j) = a_j$.

Composing game trees We consider sequential and parallel composition of game trees, for which, composing them amounts to concatenation and interleaving, respectively. Concatenating trees is more or less straightforward, since each leaf node of the first is now a root of the second tree. Interleaving trees is not the same as a tree obtained by interleaving paths from the two trees, since we wish to preserve choices made by players.

Sequential composition Suppose we are given two finite extensive form game trees $T_1 = (S_1, \Rightarrow_1, s_1^0, \widehat{\lambda}_1)$ and $T_2 = (S_2, \Rightarrow_2, s_2^0, \widehat{\lambda}_2)$. The sequential composition of T_1 and T_2 (denoted $T_1; T_2$) gives rise to a game tree $T = (S, \Rightarrow, s_0, \widehat{\lambda})$, defined as follows: $S = S_1^{NL} \cup S_2$, $s_0 = s_1^0$,

- $\widehat{\lambda}(s) = \widehat{\lambda}_1(s)$ if $s \in S_1^{NL}$ and $\widehat{\lambda}(s) = \widehat{\lambda}_2(s)$ if $s \in S_2$.
- $s \overset{a}{\Rightarrow} s'$ iff:
  - $s, s' \in S_1^{NL}$ and $s \overset{a}{\Rightarrow}_1 s'$, or
  - $s, s' \in S_2$ and $s \overset{a}{\Rightarrow}_2 s'$, or
  - $s \in S_1^{NL}, s' = s_2^0$ and there exists $s'' \in S_1^L$ such that $s \overset{a}{\Rightarrow}_1 s''$.

In other words, the game tree $T_1; T_2$ is generated by pasting the tree T_2 at all the leaf nodes of T_1. The definition of sequential composition can be extended to a set of trees $\mathcal{T}_2$ (denoted $T_1; \mathcal{T}_2$) with the interpretation that at each leaf node of T_1, a tree $T_2 \in \mathcal{T}_2$ is attached.

Parallel composition The parallel composition of T_1 and T_2 (denoted $T_1 \| T_2$) yields a set of trees. A tree $t = (S, \Rightarrow, s_0, \widehat{\lambda})$ is in the set of trees $T_1 \| T_2$ provided: $S \subseteq S_1 \times S_2$, $s_0 = (s_1^0, s_2^0)$,

– For all $(s, s') \in S$:
  - $E_T((s, s')) = E_{T_1}(s) \times s'$ and $\widehat{\lambda}(s, s') = \widehat{\lambda}_1(s)$, or
  - $E_T((s, s')) = s \times E_{T_2}(s')$ and $\widehat{\lambda}(s, s') = \widehat{\lambda}_2(s')$.

– For every edge $s_1 \overset{a}{\Rightarrow}_1 s_1'$ in T_1, there exists $s_2 \in S_2$ such that $(s_1, s_2) \overset{a}{\Rightarrow} (s_1', s_2)$ in t.

– For every edge $s_2 \overset{a}{\Rightarrow}_2 s_2'$ in T_2, there exists $s_1 \in S_1$ such that $(s_1, s_2) \overset{a}{\Rightarrow} (s_1, s_2')$ in t.

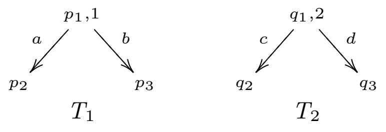


Fig. 1. atomic games

Examples Consider the trees T_1 and T_2 given in Figure 1. The sequential composition of T_1 and T_2 (denoted $T_1; T_2$) is shown in Figure 2. This is obtained by pasting the tree T_2 at all the leaf nodes of T_1.

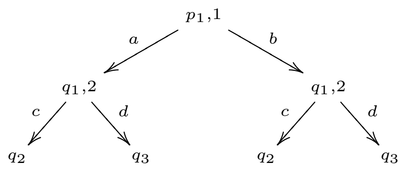
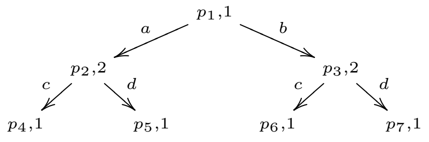


Fig. 2. $T_1; T_2$

Now consider two finite extensive form game trees T_4 and T_5 given in Figure 3. Each game is played between two players, player 2 is common in both games.

Note that we are talking about different instances of the same game (as evident from the similar game trees) played between different pairs of players with a player in common. Consider the interleaving of T_4 and T_5 where player 1 moves first in T_4, followed by 2 and 3 in T_5, and then again coming back to the game T_4, with the player 2-moves. This game constitutes a valid tree in the set of trees defined by $T_4 \| T_5$ and is shown in Figure 4.


(a) T_4

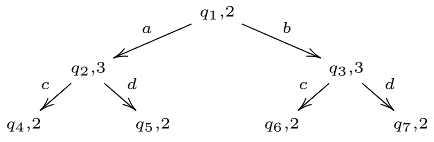

(b) T_5

Fig. 3. Atomic games

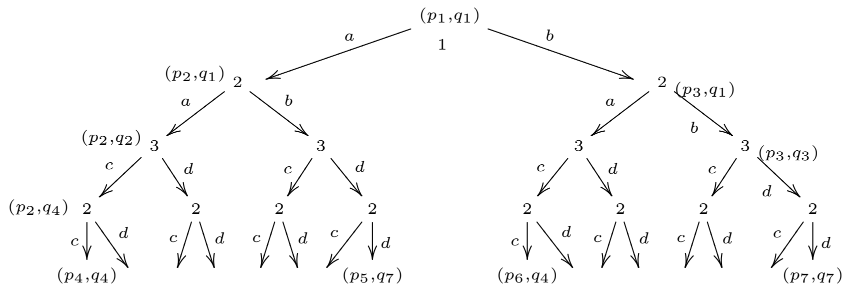


Fig. 4. Game tree T

Due to space constraints, we have not provided the names for each of the states in the parallel game tree, but they are quite clear from the context. The game starts with player 1 moving from p_1 in T_4 to p_2 or p_3. Then the play moves to the game T_5, where player 2 moves to q_2 or q_3, followed by the moves of player 3. After that, the play comes back to T_4, where player 2 moves once again.

These games clearly represent toy versions of "playing against two Grandmasters simultaneously". Players 1 and 3 can be considered as the Grandmasters, and 2 as the poor mortal. Let us now describe the copycat strategy that can be used by player 2, when the two games are played in parallel. The simultaneous game (Figure 4), starts with player 1 making the first move a, say in the game tree T_4 (from (p_1, q_1)) to move to (p_2, q_1). Player 2 then copies this move in game T_5, to move to (p_2, q_2). The game continues in T_5, with player 3 moving to (p_2, q_4), say. Player 2 then copies this move in T_4 (playing action c) to move to (p_4, q_4). This constitutes a play of the game, where player 2 copies the moves of players 1 and 3, respectively.

Evidently, if player 1 has a strategy in T_4 to achieve a certain objective, whatever be the moves of player 2, following the same strategy, player 2 can attain the same objective in T_5.

The logic For a finite set of action symbols Σ, let $\mathcal{T}(\Sigma)$ be a countable set of finite extensive form game trees over the action set Σ which is closed under subtree inclusion. That is, if $T \in \mathcal{T}(\Sigma)$ and T' is a subtree of T then $T' \in \mathcal{T}(\Sigma)$.

We also assume that for each $a \in \Sigma$, the tree consisting of the single edge labelled with a is in $\mathcal{T}(\Sigma)$. Let $\mathbb{H}$ be a countable set and h, h' range over this set. Elements of $\mathbb{H}$ are referred to in the formulas of the logic and the idea is to use them as names for extensive form game trees in $\mathcal{T}(\Sigma)$. Formally we have a map $\nu : \mathbb{H} \to \mathcal{T}(\Sigma)$ which given any name $h \in \mathbb{H}$ associates a tree $\nu(h) \in \mathcal{T}(\Sigma)$. We often abuse notation and use h to also denote $\nu(h)$ where the meaning is clear from the context.

Syntax Let P be a countable set of propositions, the syntax is given by:

$$\Gamma := h \mid g_1; g_2 \mid g_1 \cup g_2 \mid g_1 \| g_2$$

$$\Phi := p \in P \mid \neg\varphi \mid \varphi_1 \vee \varphi_2 \mid \langle g, i \rangle \varphi$$

where $h \in \mathbb{H}$ and $g \in \Gamma$.

In Γ, the atomic construct h specifies a finite extensive form game tree. Composite games are constructed using the standard dynamic logic operators along with the parallel operator. The game $g_1 \cup g_2$ denotes playing g_1 or g_2. Sequential composition is denoted by $g_1; g_2$ and $g_1 \| g_2$ denotes the parallel composition of games. The main connective $\langle g, i \rangle \varphi$ asserts at state s that a tree in g is enabled at s and that player i has a strategy subtree in it at whose leaves φ holds.

Semantics A model is a structure $M = (W, \to, \widehat{\lambda}, V)$ where W is the set of states (or game positions), $\to \subseteq W \times \Sigma \times W$ is the move relation, $V : W \to 2^P$ is a valuation function and $\widehat{\lambda} : W \to N$ is a player labelling function. These can be thought of as standard Kripke structures whose states correspond to game positions along with an additional player labelling function. An extensive form game tree can be thought of as enabled at a certain state, say s of a Kripke structure, if we can embed the tree structure in the tree unfolding of the Kripke structure rooted at s. We make this notion more precise below.

Enabling of trees For a game position $w \in W$, let T_w denote the tree unfolding of M rooted at w. We say the game h is enabled at a state w if the structure $\nu(h)$ can be embedded in T_w with respect to the enabled actions and player labelling. Formally this can be defined as follows:

Given a state w and $h \in \mathbb{H}$, let $T_w = (S_M^s, \Rightarrow_M, \widehat{\lambda}_M, s)$ and $\nu(h) = T_h = (S_h, \Rightarrow_h, \widehat{\lambda}_h, s_{h,0})$. The restriction of T_w with respect to the game tree h (denoted $T_w \upharpoonright h$) is the subtree of T_w which is generated by the structure specified by T_h. The restriction is defined inductively as follows: $T_w \upharpoonright h = (S, \Rightarrow, \widehat{\lambda}, s_0, f)$ where $f : S \to S_h$. Initially $S = \{s\}$, $\widehat{\lambda}(s) = \widehat{\lambda}_M(s)$, $s_0 = s$ and $f(s_0) = s_{h,0}$.

For any $s \in S$, let $f(s) = t \in S_h$. Let $\{a_1, \ldots, a_k\}$ be the outgoing edges of t, i.e. for all $j : 1 \leq j \leq k$, $t \overset{a_j}{\Rightarrow}_h t_j$. For each a_j, let $\{s_j^1, \ldots, s_j^m\}$ be the nodes in S_M^s such that $s \overset{a_j}{\Rightarrow}_M s_j^l$ for all $l : 1 \leq l \leq m$. Add nodes $s_j^1, \ldots, s_j^m$ to S and the edges $s \overset{a_j}{\Rightarrow} s_j^l$ for all $l : 1 \leq l \leq m$. Also set $\widehat{\lambda}(s_j^l) = \widehat{\lambda}_M(s_j^l)$ and $f(s_j^l) = t_j$.

We say that a game h is enabled at w (denoted *enabled*(h, w)) if the tree $T_w \restriction h = (S, \Rightarrow, \widehat{\lambda}, s_0, f)$ satisfies the following properties: for all $s \in S$,

– $moves(s) = moves(f(s))$,
– if $moves(s) \neq \emptyset$ then $\widehat{\lambda}(s) = \widehat{\lambda}_h(f(s))$.

We should note here that, strictly speaking, due to the introduction of f, $T_w \restriction h$ is not exactly a usual tree structure as we have above. But the understanding is clear.

Interpretation of atomic games To formally define the semantics of the logic, we need to first fix the interpretation of the compositional games constructs. In the dynamic logic approach, for each game construct g and player i we would associate a relation $R^i_g \subseteq (W \times 2^W)$ which specifies the outcome of a winning strategy for player i. However due to the ability of being able to interleave game positions, in this setting we need to keep track of the actual tree structure rather just the "input-output" relations, which is closer in spirit to what is done in process logics [17]. Thus for a game g and player i we define the relation $R^i_g \subseteq 2^{(W \times W)^*}$. For a pair $\mathbf{x} = (u, w) \in W \times W$ and a set of sequences $Y \in 2^{(W \times W)^*}$ we define $(u, w) \cdot Y = \{(u, w) \cdot \rho \mid \rho \in Y\}$. For $j \in \{1, 2\}$ we use $\mathbf{x}[j]$ to denote the j-th component of $\mathbf{x}$.

For each atomic game h and each state $u \in W$, we define $R^i_h(u)$ in a bottom-up manner in such a way that whenever h is enabled at u, $R^i_h(u)$ encodes the set of all available strategies for player i in the game h enabled at u. The collection of all such strategies that a player i can have, whenever the game h is enabled at some state $u \in W$ is given by R^i_h.

Let $h = (S, \Rightarrow, s_0, \widehat{\lambda})$ be a depth 1 tree with $moves(s_0) = \{a_1, \ldots, a_k\}$ and for all $s \neq s_0$, $moves(s) = \emptyset$. For $i \in N$ and a state $u \in W$, we define $R^i_h(u) \subseteq 2^{(W \times W)^*}$ as follows:

– If $\widehat{\lambda}(s_0) = i$ then $R^i_h(u) = \{X_j \mid$ *enabled*(h, u) and $X_j = \{(u, w_j)\}$ where $u \xrightarrow{a_j} w_j\}$.
– if $\widehat{\lambda}(s_0) \in \bar{\imath}$ then $R^i_h(u) = \{\{(u, w_j) \mid$ *enabled*(h, u) and $\exists a_j \in moves(s_0)$ with $u \xrightarrow{a_j} w_j\}\}$.

For $g \in \Gamma$, let $R^i_g = \bigcup_{w \in W} R^i_g(w)$.

For a tree $h = (S, \Rightarrow, s_0, \widehat{\lambda})$ such that $depth(h) > 1$, we define $R^i_h(u)$ as,

– if $\widehat{\lambda}(s_0) = i$ then $R^i_h(u) = \{\{(w, w) \cdot Y\} \mid \exists X \in R^i_{head(h)}$ with $(u, w) \in X, u \xrightarrow{a_j} w$ and $Y \in R^i_{h_{a_j}}\}$
– if $\widehat{\lambda}(s_0) \in \bar{\imath}$ then $R^i_h(u) = \{\{(w, w) \cdot Y \mid \exists X \in R^i_{head(h)}$ with $(u, w) \in X, u \xrightarrow{a_j} w$ and $Y \in R^i_{h_{a_j}}\}\}$.

Remark Note that a set $X \in R^i_h$ can contain sequences such as $(u, w)(v, x)$ where $w \neq v$. Thus in general sequence of pairs of states in X need not represent a subtree of T_w for some $w \in W$. We however need to include such sequences since if h is interleaved with another game tree h', a move enabled in h' could make the transition from w to v. A sequence $\varrho \in X$ is said to be legal if whenever $(u, w)(v, x)$ is a subsequence of ϱ then $w = v$. A set $X \subseteq 2^{(W \times W)^*}$ is a valid tree if for all sequence $\varrho \in X$, ϱ is legal and X is prefix closed. For X which is a valid tree we have the property that for all $\varrho, \varrho' \in X$, $\mathit{first}(\varrho)[1] = \mathit{first}(\varrho')[1]$. We denote this state by $\mathit{root}(X)$. We also use $\mathit{frontier}(X)$ to denote the frontier nodes, i.e. $\mathit{frontier}(X) = \{\mathit{last}(\varrho)[2] \mid \varrho \in X\}$.

For a game tree h, although every set $X \in R_h^i$ need not be a valid tree, we can associate a tree structure with X (denoted $\mathfrak{T}(X)$) where the edges are labelled with pairs of the form (u, w) which appears in X. Conversely given $W \times W$ edge labelled finite game tree $\mathfrak{T}$, we can construct a set $X \subseteq 2^{(W \times W)^*}$ by simply enumerating the paths and extracting the labels of each edge in the path. We denote this translation by $\mathfrak{f}(\mathfrak{T})$. We use these two translations in what follows:

Interpretation of composite games For $g \in \Gamma$ and $i \in N$, we define $R_g^i \subseteq 2^{(W \times W)^*}$ as follows:

- $R^i_{g_1 \cup g_2} = R^i_{g_1} \cup R^i_{g_2}$.
- $R^i_{g_1 ; g_2} = \{\mathfrak{f}(\mathfrak{T}(X); \mathcal{T}) \mid X \in R^i_{g_1}$ and $\mathcal{T} = \{\mathfrak{T}(X_1), \ldots, \mathfrak{T}(X_k)\}$ where $\{X_1, \ldots, X_k\} \subseteq R^i_{g_2}\}$.
- $R^i_{g_1 \| g_2} = \{\mathfrak{f}(\mathfrak{T}(X_1) \| \mathfrak{T}(X_2)) \mid X_1 \in R^i_{g_1}$ and $X_2 \in R^i_{g_2}\}$.

The truth of a formula $\varphi \in \Phi$ in a model M and a position w (denoted $M, w \models \varphi$) is defined as follows:

- $M, w \models p$ iff $p \in V(w)$.
- $M, w \models \neg\varphi$ iff $M, w \not\models \varphi$.
- $M, w \models \varphi_1 \vee \varphi_2$ iff $M, w \models \varphi_1$ or $M, w \models \varphi_2$.
- $M, w \models \langle g, i \rangle \varphi$ iff $\exists X \in R_g^i$ such that X constitutes a valid tree, $\mathit{root}(X) = w$ and for all $v \in \mathit{frontier}(X)$, $M, v \models \varphi$.

A formula φ is satisfiable if there exists a model M and a state w such that $M, w \models \varphi$.

Let h_1 and h_2 be the game trees T_4 and T_5 given in Figure 3. The tree in which the moves of players are interleaved in lock-step synchrony is one of the trees in the semantics of $h_1 \| h_2$. This essentially means that at every other stage if a depth one tree is enabled then after that the same tree structure is enabled again, except for the player labelling. Given the (finite) atomic trees, we can write a formula φ_{LS} which specifies this condition. If the tree h is a minimal one, i.e. of depth one given by $(S, \Rightarrow, s_0, \widehat{\lambda})$, φ_{LS_h} can be defined as, $\bigwedge_{a_j \in moves(s_0)}(\langle a_j \rangle \top \wedge [a_j](\wedge_{a_j \in moves(s_0)} \langle a_j \rangle \top)$.

If player 1 has a strategy (playing a, say) to achieve certain objective φ in the game h_1, player 2 can play (copy) the same strategy in h_2 to ensure φ. This phenomenon can be adequately captured in the interleaved game structure, where player 2 has a strategy (viz. playing a) to end in those states of the game $h_1 \| h_2$, where player 1 can end in h_1. So we have that, whenever h_1 and $h_1 \| h_2$ are enabled and players can move in lock-step synchrony with respect to the game h_1 (or, h_2), $\langle h_1, 1\rangle\varphi \to \langle h_1 \| h_2, 2\rangle\varphi$ holds.

Axiom system The main technical contribution of this section is a sound and complete axiom system. For $a \in \Sigma$ and $i \in N$, let T^i_a be the tree defined as: $T^i_a = (S, \Rightarrow, s_0, \widehat{\lambda})$ where $S = \{s_0, s_1\}$, $s_0 \overset{a}{\Rightarrow} s_1$, $\widehat{\lambda}(s_0) = i$ and $\widehat{\lambda}(s_1) \in N$. Let t^i_a be the name denoting this tree, i.e. $\nu(t^i_a) = T^i_a$. For each $a \in \Sigma$ we define,

– $\langle a\rangle\varphi = \bigwedge_{i \in N}(\mathbf{turn}_i \to \langle t^i_a, i\rangle\varphi)$.

From the semantics it is easy to see that we get the standard interpretation for $\langle a\rangle\varphi$, i.e. $\langle a\rangle\varphi$ holds at a state u iff there is a state w such that $u \xrightarrow{a} w$ and φ holds at w.

Enabling of trees The crucial observation is that the property of whether a game is enabled can be described by a formula of the logic. Formally, for $h \in \mathbb{H}$ such that $\nu(h) = (S, \Rightarrow, s_0, \widehat{\lambda})$ and $moves(s_0) \neq \emptyset$ and an action $a \in moves(s_0)$, let h_a be the subtree of T rooted at a node s' with $s_0 \overset{a}{\Rightarrow} s'$. The formula $h^{\checkmark}$ (defined below) is used to express the fact that the tree structure $\nu(h)$ is enabled and $head^{\checkmark}_h$ to express that $head(\nu(h))$ is enabled. This is defined as,

– If $\nu(h)$ is atomic then $h^{\checkmark} = \top$ and $head^{\checkmark}_h = \top$.
– If $\nu(h)$ is not atomic and $\widehat{\lambda}(s_0) = i$ then
  • $h^{\checkmark} = \mathbf{turn}_i \wedge (\bigwedge_{a_j \in moves(s_0)}(\langle a_j\rangle\top \wedge [a_j]h^{\checkmark}_{a_j}))$.
  • $head^{\checkmark}_h = \mathbf{turn}_i \wedge (\bigwedge_{a_j \in moves(s_0)} \langle a_j\rangle\top)$.

Due to the ability to interleave choices of players, we also need to define for a composite game expression g, the initial (atomic) game of g and the game expression generated after playing the initial atomic game (or in other words the residue). We make this notion precise below:

Definition of init

– $init(h) = \{h\}$ for $h \in \mathrm{G}$
– $init(g_1; g_2) = init(g_1)$ if $g_1 \neq \epsilon$ else $init(g_2)$.
– $init(g_1 \cup g_2) = init(g_1) \cup init(g_2)$.
– $init(g_1 \| g_2) = init(g_1) \cup init(g_2)$.

Definition of residue

– $h \backslash h = \epsilon$ and $\epsilon \backslash h = \epsilon$.
– $(g_1; g_2)\backslash h = \begin{cases} (g_1\backslash h); g_2 & \text{if } \; g_1 \neq \epsilon. \\ (g_2\backslash h) & \text{otherwise.} \end{cases}$
– $(g_1 \cup g_2)\backslash h = \begin{cases} (g_1\backslash h) \cup (g_2\backslash h) & \text{if } \; h \in init(g_1) \text{ and } h \in init(g_2). \\ g_1\backslash h & \text{if } \; h \in init(g_1) \text{ and } h \notin init(g_2). \\ g_2\backslash h & \text{if } \; h \in init(g_2) \text{ and } h \notin init(g_1). \end{cases}$

– $(g_1 \| g_2)\backslash h = \begin{cases} (g_1\backslash h \| g_2) \cup (g_1 \| g_2\backslash h) & \text{if } h \in \mathit{init}(g_1) \text{ and } h \in \mathit{init}(g_2). \\ (g_1\backslash h \| g_2) & \text{if } h \in \mathit{init}(g_1) \text{ and } h \notin \mathit{init}(g_2). \\ (g_1 \| g_2\backslash h) & \text{if } h \in \mathit{init}(g_2) \text{ and } h \notin \mathit{init}(g_1). \end{cases}$

The translation used to express the property of enabling of trees in terms of standard *PDL* formulas also suggest that the techniques developed for proving completeness of *PDL* can be applied in the current setting. We base our axiomatization of the logic on the "reduction axioms" methodology of dynamic logic. The most interesting reduction axiom in our setting would naturally involve the parallel composition operator. Intuitively, for game expressions g_1, g_2, a formula φ and a player $i \in N$ the reduction axiom for $\langle g_1 \| g_2, i\rangle\varphi$ need to express the following properties:

– There exists an atomic tree $h \in \mathit{init}(g_1 \| g_2)$ such that $\mathit{head}(\nu(h))$ is enabled.
– Player i has a strategy in $\mathit{head}(\nu(h))$ which when composed with a strategy in the residue ensures φ. We use $\mathit{comp}^i(h, g_1, g_2, \varphi)$ to denote this property and formally define it inductively as follows:

Suppose $h = (S, \Rightarrow, s_0, \widehat{\lambda})$ where $A = \mathit{moves}(s_0) = \{a_1, \dots, a_k\}$.

– If $h \in \mathit{init}(g_1)$, $h \in \mathit{init}(g_2)$ and
  • $\widehat{\lambda}(s_0) = i$ then $\mathit{comp}^i(h, g_1, g_2, \varphi) = \bigvee_{a_j \in A}(\langle a_j\rangle\langle(h_{a_j}; (g_1\backslash h))\|g_2\rangle\varphi \vee \langle a_j\rangle\langle g_1\|(h_{a_j}; (g_2\backslash h))\rangle\varphi)$.
  • $\widehat{\lambda}(s_0) \in \bar{\imath}$ then $\mathit{comp}^i(h, g_1, g_2, \varphi) = \bigwedge_{a_j \in A}([a_j]\langle(h_{a_j}; (g_1\backslash h))\|g_2\rangle\varphi \vee [a_j]\langle g_1\|(h_{a_j}; (g_2\backslash h))\rangle\varphi)$.
– If $h \in \mathit{init}(g_1)$, $h \notin \mathit{init}(g_2)$ and
  • $\widehat{\lambda}(s_0) = i$ then $\mathit{comp}^i(h, g_1, g_2, \varphi) = \bigvee_{a_j \in A}(\langle a_j\rangle\langle(h_{a_j}; (g_1\backslash h))\|g_2\rangle\varphi)$.
  • $\widehat{\lambda}(s_0) \in \bar{\imath}$ then $\mathit{comp}^i(h, g_1, g_2, \varphi) = \bigwedge_{a_j \in A}([a_j]\langle(h_{a_j}; (g_1\backslash h))\|g_2\rangle\varphi)$.
– if $h \in \mathit{init}(g_2)$, $h \notin \mathit{init}(g_1)$ and
  • $\widehat{\lambda}(s_0) = i$ then $\mathit{comp}^i(h, g_1, g_2, \varphi) = \bigvee_{a_j \in A}(\langle a_j\rangle\langle g_1\|(h_{a_j}; (g_2\backslash h))\rangle\varphi)$.
  • $\widehat{\lambda}(s_0) \in \bar{\imath}$ then $\mathit{comp}^i(h, g_1, g_2, \varphi) = \bigwedge_{a_j \in A}([a_j]\langle g_1\|(h_{a_j}; (g_2\backslash h))\rangle\varphi)$.

Note that the semantics for parallel composition allows us to interleave subtrees of g_2 within g_1 (and vice versa). Therefore in the definition of comp^i at each stage after an action a_j, it is important to perform the sequential composition of the subtree h_{a_j} with the residue of the game expression.

The axiom schemes

(A1) Propositional axioms:
  (a) All the substitutional instances of tautologies of PC.
  (b) $\mathbf{turn}_i \leftrightarrow \bigwedge_{j \in \bar{\imath}} \neg\mathbf{turn}_j$.
(A2) Axiom for single edge games:
  (a) $\langle a\rangle(\varphi_1 \vee \varphi_2) \leftrightarrow \langle a\rangle\varphi_1 \vee \langle a\rangle\varphi_2$.
  (b) $\langle a\rangle\mathbf{turn}_i \rightarrow [a]\mathbf{turn}_i$.
(A3) Dynamic logic axioms:

(a) $\langle g_1 \cup g_2, i\rangle\varphi \leftrightarrow \langle g_1, i\rangle\varphi \vee \langle g_2, i\rangle\varphi$.
(b) $\langle g_1; g_2, i\rangle\varphi \leftrightarrow \langle g_1, i\rangle\langle g_2, i\rangle\varphi$.
(c) $\langle g_1 \| g_2, i\rangle\varphi \leftrightarrow \bigvee_{h \in init(g_1 \| g_2)} head_h^{\checkmark} \wedge comp^i(h, g_1, g_2, \varphi)$.

(A4) $\langle h, i\rangle\varphi \leftrightarrow h^{\checkmark} \wedge \downarrow_{(h,i,\varphi)}$.

For $h \in \mathbb{H}$ with $\nu(h) = T = (S, \Rightarrow, s_0, \widehat{\lambda})$ we define $\downarrow_{(h,i,\varphi)}$ as follow:

– $\downarrow_{(h,i,\varphi)} = \begin{cases} \varphi & \text{if } moves(s_0) = \emptyset. \\ \bigvee_{a \in \Sigma} \langle a\rangle\langle h_a, i\rangle\varphi & \text{if } moves(s_0) \neq \emptyset \text{ and } \widehat{\lambda}(s_0) = i. \\ \bigwedge_{a \in \Sigma} [a]\langle h_a, i\rangle\varphi & \text{if } moves(s_0) \neq \emptyset \text{ and } \widehat{\lambda}(s_0) \in \bar{\imath}. \end{cases}$

Inference rules

$$(MP)\ \frac{\varphi, \quad \varphi \rightarrow \psi}{\psi} \qquad (NG)\ \frac{\varphi}{[a]\varphi}$$

Axioms (A1) and (A2) are self explanatory. Axiom (A3) constitutes the reduction axioms for the compositional operators, and Axiom (A4) is the atomic game axiom. Note that unlike in *PDL* sequential composition in our setting corresponds to composition over trees. The details of the completeness proof can be found in [14].

5 Making strategies explicit

In this section, we discuss strategizing in large extensive form games (game trees) by resource limited players. Such players rely on 'local' and 'partial' plans which can be composed to form more 'global' plans. The notion of structures in strategies become meaningful, leading to logical studies of structured strategies.

Dually, one could also consider the game tree as obtained by composition from a collection of 'small' game trees constituting subgames and strategies as 'complete' plans on them to ensure local outcomes (which may be the initiation of desired subgames).

In the following we discuss a logical language in which strategies are partial plans, have compositional structure and we reason about agents employing such strategies to achieve desired outcomes. We also consider functional (total) strategies in finite subgames, and lift them over to strategies in structured games. For more details on the results mentioned in this section see [41,15].

Before proceeding any further, we should mention here that the explicit study of actions and strategies have been going on for a while in various logical frameworks, and one can get acquainted with the different approaches in coalition logics [5], temporal logics [20,43], first and second order logics [7,35]. A point of departure in all these frameworks from what we are going to present below is that, these frameworks consider strategies at an atomic level, giving different 'names' to different strategies or actions, whereas we will concentrate on studying composite structure in strategies.

Preliminary notions Extensive form games are a natural model for representing *finite games* in an explicit manner. In this model, the game is represented as a finite tree where the nodes of the tree corresponds to the game positions and edges correspond to moves of players. The leaf nodes are labelled with payoffs obtained by players. For formal definitions we refer to Section 4.4. For simplicity, we will consider two person games only. The results can similarly be proved for n person games.

5.1 Strategy specifications

Similar to what has been presented in Section 3.2, we conceive of strategy specifications as being built up from atomic ones using some grammar. The atomic case specifies, for a player, what conditions she tests for before making a move. These constitute positional strategies and the pre-condition for the move depends on observables that hold at the current game position and some finite look-ahead that each player can perform in terms of the structure of the game tree. In addition to the usage of past time formulas to represent the observables (cf. Section 3.2), another method is to state these pre-conditions as future time formulas of a simple action indexed tense logic over the observables (to facilitate describing finite look-ahead). The structured strategy specifications are then built from atomic ones using connectives.

Below, for any countable set X, let $BF(X)$ be sets of formulas given by the following syntax:

$$BF(X) := x \in X \mid \neg\psi \mid \psi_1 \vee \psi_2 \mid \langle a\rangle\psi$$

where $a \in \Sigma$.

Formulas in $BF(X)$ are interpreted at game positions. The formula $\langle a\rangle\psi$ talks about one step in the future. It asserts the existence of an a edge after which ψ holds. Note that future time assertions up to any bounded depth can be coded by iteration of this construct. The "future free" fragment of $BF(X)$ are the Boolean formulas over X, we denote this fragment by $Bool(X)$.

Syntax Let $P^i = \{p_0^i, p_1^i, \ldots\}$ be a (non-empty) countable set of observables for $i \in N$ and $P = \cup_{i\in N} P^i$. The syntax of strategy specifications is given by:

$$Strat^i(P^i) := [\psi \mapsto a]^i \mid \sigma_1 + \sigma_2 \mid \sigma_1 \cdot \sigma_2 \mid \pi \Rightarrow \sigma_1.$$

where $\psi \in BF(P^i)$ and $\pi \in Strat^{\bar{\imath}}(P^i \cap P^{\bar{\imath}})$. The intuitive interpretations of these constructs can be found in Section 3.2.

Semantics Let $M = (T, V)$ where $T = (S, \Rightarrow, s_0, \widehat{\lambda})$ is an extensive form game tree and $V : S \to 2^P$ a valuation function. The truth of a formula $\psi \in BF(P)$ at the state s, denoted $M, s \models \psi$ is defined as follows:

- $M, s \models p$ iff $p \in V(s)$.
- $M, s \models \neg\psi$ iff $M, s \not\models \psi$.
- $M, s \models \psi_1 \vee \psi_2$ iff $M, s \models \psi_1$ or $M, s \models \psi_2$.
- $M, s \models \langle a\rangle\psi$ iff there exists an s' such that $s \overset{a}{\Rightarrow} s'$ and $M, s' \models \psi$.

Here, strategy specifications are interpreted on strategy trees of T. We assume the presence of two special propositions $\mathbf{turn}_1$ and $\mathbf{turn}_2$ that specifies which player's turn it is to move. We also assume the existence of a special proposition *leaf* which holds at the terminal nodes. Formally, we assume that the valuation function satisfies the property:

- for all $i \in N$, $\mathbf{turn}_i \in V(s)$ iff $\widehat{\lambda}(s) = i$.
- $leaf \in V(s)$ iff $moves(s) = \emptyset$.

Recall that a strategy μ of player i can be viewed as a subtree $T_\mu = (S_\mu, \Rightarrow_\mu, s_0, \widehat{\lambda}_\mu)$ of T. Let V_μ denote the restriction of the valuation function V to S_μ and $\rho_{s_0}^s : s_0 a_0 s_1 \cdots s_m = s$ be the unique path in μ from the root node s_0 to s. For a strategy specification $\sigma \in Strat^i(P^i)$, we define the notion of μ conforming to σ (denoted $(T_\mu, V_\mu) \models \sigma$) as follows:

- $(T_\mu, V_\mu) \models \sigma$ iff for every player i node $s \in S_\mu$, we have $\rho_{s_0}^s, s \models \sigma$.

where we define (similar to the definition given in Section 3.2, with some notational modifications) $\rho_{s_0}^s, s_j \models \sigma$ for any player i node s_j in $\rho_{s_0}^s$ as,

- $\rho^s_{s_0}, s_j \models_i [\psi \mapsto a]^i$ iff $(T_\mu, V_\mu), s_j \models \psi$ implies $out_{\rho^s_{s_0}}(s_j) = a$.
- $\rho^s_{s_0}, s_j \models_i \sigma_1 + \sigma_2$ iff $\rho^s_{s_0}, s_j \models \sigma_1$ or $\rho^s_{s_0}, s_j \models \sigma_2$.
- $\rho^s_{s_0}, s_j \models_i \sigma_1 \cdot \sigma_2$ iff $\rho^s_{s_0}, s_j \models \sigma_1$ and $\rho^s_{s_0}, s_j \models \sigma_2$.
- $\rho^s_{s_0}, s_j \models_i \pi \Rightarrow \sigma_1$ iff (if for all player $\bar{\imath}$ nodes $s_k \in \rho^s_{s_0}$ such that $k \leq j$, $\rho^s_{s_0}, s_k \models_{\bar{\imath}} \pi$) then $\rho^s_{s_0}, s_j \models_i \sigma_1$.

Above, $\psi \in BF(P^i)$ and for all $i : 0 \leq i < m$, $out_{\rho^s_{s_0}}(s_i) = a_i$ and $out_\mu(s)$ is the unique outgoing edge in T_μ at s. Recall that s is a player i node and therefore by definition there is a unique outgoing edge at s.

5.2 A strategy logic

We now discuss how we may embed structured strategies in a formal logic. Formulas of the logic are built up using structured strategy specification. The formulas describe the game arena in a standard modal logic, and in addition specify the result of a player following a particular strategy at a game position, to choose a specific move a. Using these formulas one can specify how a strategy helps to eventually *win* (ensure) an outcome β.

Syntax The syntax of the logic is given by:

$$\mathcal{L}_s := p \in P \mid (\sigma)_i : c \mid \neg\alpha \mid \alpha_1 \vee \alpha_2 \mid \langle a \rangle\alpha \mid \langle \overline{a} \rangle\alpha \mid \diamondminus\alpha \mid \sigma \leadsto_i \beta.$$

where $a, c \in \Sigma$, $\sigma \in Strat^i(P^i)$, $\beta \in Bool(P^i)$. The derived connectives $\wedge$ and $\rightarrow$ are defined as usual. Let $\boxminus\alpha = \neg\diamondminus\neg\alpha$, $\langle P \rangle\alpha = \bigvee_{a \in \Sigma} \langle \overline{a} \rangle\alpha$, $[a]\alpha = \neg\langle a \rangle\neg\alpha$, $\bigcirc\alpha = \bigvee_{a \in \Sigma} \langle a \rangle\alpha$ and $\odot\,\alpha = \neg\bigcirc\neg\alpha$.

Intuitively, the formula $(\sigma)_i : c$ asserts, at any game position, that the strategy specification σ for player i allows the move c to be played at that position. The formula $\sigma \leadsto_i \beta$ says that from this position, choosing any move allowed by the strategy specification σ for player i ensures the outcome β. These two modalities constitute the main constructs of the logic.

Semantics Models of the logic are of the form $M = (T, V)$ where $T = (S, \Rightarrow, s_0, \widehat{\lambda})$ is an extensive form game tree and $V : S \rightarrow 2^P$ is a valuation function. As mentioned earlier, we require that the valuation function satisfies the condition:

- For all $s \in S$ and $i \in N$, $\mathbf{turn}_i \in V(s)$ iff $\widehat{\lambda}(s) = i$.

For the purpose of defining the logic it is convenient to define the notion of the set of moves enabled by a strategy specification σ at a game position s (denoted $\sigma(s)$). Given a model $M = (T, V)$ where $T = (S, \Rightarrow, s_0, \widehat{\lambda})$, a node $s \in S$ and a strategy specification $\sigma \in Strat^i(P^i)$ we define $\sigma(s)$ as follows:

– $[\psi \mapsto a]^i(s) = \begin{cases} \{a\} \text{ if } \widehat{\lambda}(s) = i,\ M, s \models \psi \text{ and } a \in moves(s). \\ \emptyset \quad \text{if } \widehat{\lambda}(s) = i,\ M, s \models \psi \text{ and } a \notin moves(s). \\ \Sigma \quad \text{otherwise.} \end{cases}$
– $(\sigma_1 + \sigma_2)(s) = \sigma_1(s) \cup \sigma_2(s)$.
– $(\sigma_1 \cdot \sigma_2)(s) = \sigma_1(s) \cap \sigma_2(s)$.
– $(\pi \Rightarrow \sigma)(s) = \begin{cases} \sigma(s) \text{ if } \forall j : 0 \leq j < m,\ a_j \in \pi(s_j). \\ \Sigma \quad \text{otherwise.} \end{cases}$

We say that a path $\rho_s^{s'} : s = s_1 \overset{a_1}{\Rightarrow} s_2 \cdots \overset{a_{m-1}}{\Rightarrow} s_m = s'$ in T conforms to σ if $\forall j : 1 \leq j < m,\ a_j \in \sigma(s_j)$. When the path constitutes a proper play, i.e. when $s = s_0$, we say that the play conforms to σ.

For a game tree T and a node $s \in S$, let T_s denote the tree which consists of the unique path $\rho_{s_0}^s$ and the subtree rooted at s. For a strategy specification $\sigma \in Strat^i(P^i)$, we define $T_s \upharpoonright \sigma = (S_\sigma, \Rightarrow_\sigma, s_0, \widehat{\lambda}_\sigma)$ to be the least subtree of T_s which contains the unique path from s_0 to s and satisfies the property: for every $s_1 \in S_\sigma$,

– if $\widehat{\lambda}_\sigma(s_1) = i$ then for all s_2 with $s_1 \overset{a}{\Rightarrow} s_2$ and $a \in \sigma(s_1)$ we have $s_1 \overset{a}{\Rightarrow}_\sigma s_2$ and $\widehat{\lambda}_\sigma(s_2) = \widehat{\lambda}(s_2)$.
– if $\widehat{\lambda}_\sigma(s_1) = \overline{\imath}$ then for all s_2 with $s_1 \overset{a}{\Rightarrow} s_2$ we have $s_1 \overset{a}{\Rightarrow}_\sigma s_2$ and $\widehat{\lambda}_\sigma(s_2) = \widehat{\lambda}(s_2)$.

The truth of a formula $\alpha \in \mathcal{L}_s$ in a model M and position s (denoted $M, s \models \alpha$) is defined by induction on the structure of α, as usual. Let $\rho_{s_0}^s$ be $s_0 \overset{a_0}{\Rightarrow} s_1 \cdots \overset{a_{m-1}}{\Rightarrow} s_m = s$.

– $M, s \models p$ iff $p \in V(s)$.
– $M, s \models \neg\alpha$ iff $M, s \not\models \alpha$.
– $M, s \models \alpha_1 \vee \alpha_2$ iff $M, s \models \alpha_1$ or $M, s \models \alpha_2$.
– $M, s \models \langle a \rangle \alpha$ iff there exists s' such that $s \overset{a}{\Rightarrow} s'$ and $M, s' \models \alpha$.
– $M, s \models \langle \overline{a} \rangle \alpha$ iff $m > 0$, $a = a_{m-1}$ and $M, s_{m-1} \models \alpha$.
– $M, s \models \diamondminus \alpha$ iff there exists $j : 0 \leq j \leq m$ such that $M, s_j \models \alpha$.
– $M, s \models (\sigma)_i : c$ iff $c \in \sigma(s)$.
– $M, s \models \sigma \leadsto_i \beta$ iff for all s' such that $s \Rightarrow_\sigma^* s'$ in $T_s \upharpoonright \sigma$, we have $M, s' \models \beta \wedge (\mathbf{turn}_i \wedge \neg leaf \to enabled_\sigma)$.

where $enabled_\sigma = \bigvee_{a \in \Sigma} (\langle a \rangle \top \wedge (\sigma)_i : a)$ and $\Rightarrow_\sigma^*$ denotes the reflexive, transitive closure of $\Rightarrow_\sigma$.

Figure 5 illustrates the semantics of $\sigma \leadsto_1 \beta$. It says, for any 1 node β is ensured by playing according to σ; for a 2 node, all actions should ensure β.

The notions of satisfiablility and validity can be defined in the standard way. A formula α is satisfiable iff there exists a model M and s such that $M, s \models \alpha$. A formula α is said to be valid iff for all models M and for all nodes s, we have $M, s \models \alpha$.

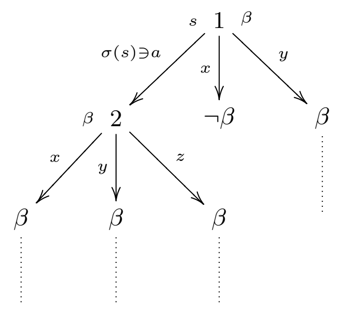


Fig. 5. Interpretation of $\sigma \rightsquigarrow_i \beta$

5.3 Axiom system

We now present an axiomatization of the valid formulas of the logic. We find the following abbreviations useful:

- $root = \neg\langle P\rangle\top$ defines the root node to be one that has no predecessors.
- $\delta_i^\sigma(a) = \mathbf{turn}_i \wedge (\sigma)_i : a$ denotes that move "a" is enabled by σ at an i node.
- $inv_i^\sigma(a,\beta) = (\mathbf{turn}_i \wedge (\sigma)_i : a) \rightarrow [a](\sigma \rightsquigarrow_i \beta)$ denotes the fact that after an "a" move by player i which conforms to σ, $\sigma \rightsquigarrow_i \beta$ continues to hold.
- $inv_{\bar{\imath}}^\sigma(\beta) = \mathbf{turn}_{\bar{\imath}} \rightarrow \bigodot(\sigma \rightsquigarrow_i \beta)$ says that after any move of $\bar{\imath}$, $\sigma \rightsquigarrow_i \beta$ continues to hold.
- $conf_\pi = \boxminus(\langle\overline{a}\rangle\mathbf{turn}_{\bar{\imath}} \rightarrow \langle\overline{a}\rangle(\pi)_{\bar{\imath}} : a)$ denotes that all opponent moves in the past conform to π.

The axiom schemes

(A1) Propositional axioms:
  (a) All the substitutional instances of tautologies of PC.
  (b) $\mathbf{turn}_i \leftrightarrow \neg\mathbf{turn}_{\bar{\imath}}$.

(A2) (a) $[a](\alpha_1 \rightarrow \alpha_2) \rightarrow ([a]\alpha_1 \rightarrow [a]\alpha_2)$
  (b) $[\overline{a}](\alpha_1 \rightarrow \alpha_2) \rightarrow ([\overline{a}]\alpha_1 \rightarrow [\overline{a}]\alpha_2)$

(A3) (a) $\langle a\rangle\alpha \rightarrow [a]\alpha$
  (b) $\langle\overline{a}\rangle\alpha \rightarrow [\overline{a}]\alpha$
  (c) $\langle\overline{a}\rangle\top \rightarrow \neg\langle\overline{b}\rangle\top$ for all $b \neq a$

(A4) (a) $\alpha \rightarrow [a]\langle\overline{a}\rangle\alpha$
  (b) $\alpha \rightarrow [\overline{a}]\langle a\rangle\alpha$

(A5) (a) $\diamondminus\, root$
  (b) $\boxminus\alpha \leftrightarrow (\alpha \wedge [P]\boxminus\alpha)$

(A6) (a) $\langle a\rangle\top \rightarrow ([\psi \mapsto a]^i)_i : a$ for all $a \in \Sigma$
  (b) $(\mathbf{turn}_i \wedge \psi \wedge ([\psi \mapsto a]^i)_i : a) \rightarrow \langle a\rangle\top$

(c) $\mathbf{turn}_i \wedge ([\psi \mapsto a]^i)_i : c \leftrightarrow \neg\psi$ for all $a \neq c$

(A7) (a) $(\sigma_1 + \sigma_2)_i : c \leftrightarrow (\sigma_1)_i : c \vee (\sigma_2)_i : c$
(b) $(\sigma_1 \cdot \sigma_2)_i : c \leftrightarrow (\sigma_1)_i : c \wedge (\sigma_2)_i : c$
(c) $(\pi \Rightarrow \sigma)_i : c \leftrightarrow \mathit{conf}_\pi \rightarrow (\sigma)_i : c$

(A8) $\sigma \leadsto_i \beta \rightarrow (\beta \wedge \mathit{inv}_i^\sigma(a,\beta) \wedge \mathit{inv}_{\bar{\imath}}^\sigma(\beta) \wedge (\neg \mathit{leaf} \rightarrow \mathit{enabled}_\sigma))$

Inference rules

$$(MP)\ \frac{\alpha, \quad \alpha \rightarrow \beta}{\beta} \qquad (NG)\ \frac{\alpha}{[a]\alpha}$$

$$(\mathit{Ind\text{-}past})\ \frac{\alpha \rightarrow [P]\alpha}{\alpha \rightarrow \boxminus\alpha}$$

$$(\mathit{Ind} \leadsto)\ \frac{\bigwedge_{a\in\Sigma}(\alpha \wedge \delta_i^\sigma(a) \rightarrow [a]\alpha), \quad \alpha \wedge \mathbf{turn}_{\bar{\imath}} \rightarrow \bigodot\alpha, \quad \alpha \wedge \neg\mathit{leaf} \rightarrow \mathit{enabled}_\sigma, \quad \alpha \rightarrow \beta}{\alpha \rightarrow \sigma \leadsto_i \beta}$$

Axioms (A1) give the propositional logic axioms with (b) corresponding to the fact that only one player can move at a node. Axioms (A2) - (A4) provide properties of the action operators. Axioms (A2) give the Kripke axiom for the operators $[a]$ and $[\overline{a}]$. Axioms (A3) and (A4) give some consistency conditions that are applied to the game tree model, e.g. (A3)(a) says that if one 'a' move from a certain state, s (say) reaches a state where 'p' holds, then it holds at all states reachable from s by the 'a' move. Axioms (A5) describe properties of the past modality, whereas (A6) and (A7) describe the semantics of strategy specifications. The rule $Ind \leadsto$ illustrates the new kind of reasoning in the logic. It says that to infer that the formula $\sigma \leadsto_i \beta$ holds in all reachable states, β must hold at the asserted state and

- for a player i node after every move which conforms to σ, β continues to hold.
- for a player $\bar{\imath}$ node after every enabled move, β continues to hold.
- player i does not get stuck by playing σ.

For a proof of completeness of the above axiom system see [15]. Given a finite presentation of the model M and a formula α it is possible to construct a Büchi tree automaton which accepts M iff $M \models \alpha$ (details can be found in [41], pages 54-62). Thus the truth checking (or the model checking) question of the logic remains decidable.

5.4 Compositional games

In order to express complex strategizing notions, it is also useful to compose game-strategy pairs rather than to treat game composition and strategic analysis as independent entities (see [15] for a detailed discussion). In this section we formalize this notion of composition. In fact we work with a more general framework of game-outcome pairs. A game-outcome pair in effect defines the functional strategies which ensure the specified outcome.

Syntax for extensive form game trees Let *Nodes* be a countable set. The syntax for specifying finite extensive form game trees is given by:

$$\mathbb{G}(Nodes) := (i, x) \mid \Sigma_{a_m \in J}((i, x), a_m, t_{a_m})$$

where $i \in N$, $x \in Nodes$, $J \subseteq \Sigma$, and $t_{a_m} \in \mathbb{G}(Nodes)$.

Given $h \in \mathbb{G}(Nodes)$ we define the tree T_h generated by h inductively as follows.

- $h = (i, x)$: $T_h = (S_h, \Rightarrow_h, \widehat{\lambda}_h, s_x)$ where $S_h = \{s_x\}$, $\widehat{\lambda}_h(s_x) = i$.
- $h = ((i, x), a_1, t_{a_1}) + \cdots + ((i, x), a_k, t_{a_k})$: Inductively we have trees $T_1, \ldots T_k$ where for $j : 1 \leq j \leq k$, $T_j = (S_j, \Rightarrow_j, \widehat{\lambda}_j, s_{j,0})$. Define $T_h = (S_h, \Rightarrow_h, \widehat{\lambda}_h, s_x)$ where
  - $S_h = \{s_x\} \cup S_{T_1} \cup \ldots \cup S_{T_k}$.
  - $\widehat{\lambda}_h(s_x) = i$ and for all j, for all $s \in S_{T_j}$, $\widehat{\lambda}_h(s) = \widehat{\lambda}_j(s)$.
  - $\Rightarrow_h = \bigcup_{j:1 \leq j \leq k}(\{(s_x, a_j, s_{j,0})\} \cup \Rightarrow_j)$.

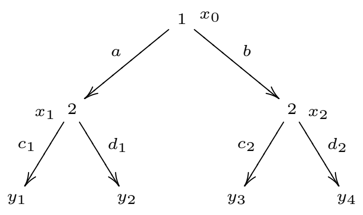


Fig. 6. Extensive form game tree

As an example, consider the extensive form game tree shown in Figure 6. The nodes are labelled with turns of players and edges with the actions. The syntactic representation of this tree can be given as follows:

- $h = ((1, x_0), a, t_1) + ((1, x_0), b, t_2)$ where
  - $t_1 = ((2, x_1), c_1, (2, y_1)) + ((2, x_1), d_1, (2, y_2))$ and
  - $t_2 = ((2, x_2), c_2, (2, y_3)) + ((2, x_2), d_2, (2, y_4))$.

Syntax for game-strategy logic Let P be a countable set of propositions, the syntax of the logic is given by:

$$\Gamma := (h, \beta) \mid g_1; g_2 \mid g_1 \cup g_2 \mid g^*$$

$$\mathcal{L}_c := p \in P \mid \neg\alpha \mid \alpha_1 \vee \alpha_2 \mid \langle g, i \rangle \alpha$$

where $h \in \mathbb{G}(Nodes)$, $\beta \in Bool(P)$ and $g \in \Gamma$.

Intuitively, the connective $\langle g, i \rangle \alpha$ asserts at state w that α holds at the leaves of a strategy tree g for player i enabled at w.

Semantics Models of the logic are pairs $M = (T, V)$ where $T = (S, \Rightarrow, s_0, \widehat{\lambda})$ is an extensive form game tree and $V : S \to 2^P$ is a valuation function. To formally define the truth of a formula in a model we need to first formalise the notion of when a game is enabled at a state of the model, similar to what we have done in Section 4.4.

Enabling of trees. For $w \in S$, let T_w denote the subtree of M starting at w. We say the game g is enabled at a state w if the structure g can be embedded in T_w with respect to the enabled actions and player labelling. Since M needs not be deterministic, there could be multiple embeddings, and therefore we work with the maximal embedding (denoted $T_w \upharpoonright g$) and this is the game tree under consideration. Formally this can be defined as follows:

Given a state w and $g \in \mathbb{G}(\mathit{Nodes})$, let $T_w = (S_M^w, \Rightarrow_M, s_w, \widehat{\lambda}_M)$ and $T_g = (S_g, \Rightarrow_g, s_{g,0}, \widehat{\lambda}_g)$. The restriction of T_w with respect to the game g (denoted $T_w \upharpoonright g$) is the subtree of T_w which is generated by the structure specified by T_g. The restriction is defined inductively as follows: $T_w \upharpoonright g = (S, \Rightarrow, s_0, \widehat{\lambda}, f)$ where $f : S \to S_g$. Initially $S = \{s_w\}$, $\widehat{\lambda}(s_w) = \widehat{\lambda}_M(s_w)$, $s_0 = s_w$ and $f(s_w) = s_{g,0}$.

For any $s \in S$, let $f(s) = t \in S_g$. Let $\{a_1, \ldots, a_k\}$ be the outgoing edges of t, i.e. for all $j : 1 \le j \le k$, $t \overset{a_j}{\Rightarrow}_g t_j$. For each a_j, let $\{s_j^1, \ldots, s_j^m\}$ be the nodes in S_M^w such that $s \overset{a_j}{\Rightarrow}_M s_j^l$ for all $l : 1 \le l \le m$. Add nodes $s_j^1, \ldots, s_j^m$ to S and the edges $s \overset{a_j}{\Rightarrow} s_j^l$ for all $l : 1 \le l \le m$. Also set $\widehat{\lambda}(s_j^l) = \widehat{\lambda}_M(s_j^l)$ and $f(s_j^l) = t_j$.

We say that a game h is enabled at w (denoted $\mathit{enabled}(h, w)$) if the tree $T_w \upharpoonright g = (S, \Rightarrow, s_0, \widehat{\lambda}, f)$ satisfies the following property: for all $s \in S$,

- $\vec{s} = \overrightarrow{f(s)}$,
- if $\vec{s} \neq \emptyset$ then $\widehat{\lambda}(s) = \widehat{\lambda}_g(f(s))$.

Having defined the notion of an atomic tree being enabled at a state, the next step is to "interpret" compositional games. Intuitively with each compositional game $g \in \Gamma$ and player $i \in N$ we associate the collection of states that player can ensure by employing a strategy in g. Formally we define sets $R_g^i \subseteq W \times 2^W$ by induction on the structure of g. For a game tree T, let $\Omega^i(T)$ denote the set of strategies of player i on the game tree T and $\mathit{frontier}(T)$ denote the set of all leaf nodes of T.

Atomic pair For an atomic pair $g = (h, \beta)$ and $i \in N$, we define R_g^i as follows:

- $R^i_{(h,\beta)} = \{(u, X) \mid \mathit{enabled}(h, u)$ and $\exists \mu \in \Omega^i(T_u \upharpoonright h)$ such that $\mathit{frontier}(\mu) = X$ and $\forall s \in X,\ M, s \models \beta\}$.

where for a Boolean formula β, we have the standard interpretation for $M, s \models \beta$.

Composition Sequential composition and choice is interpreted as follows:

- $R^i_{g_1;g_2} = \{(u, X) \mid \exists Y \subseteq W$ such that $(u, Y) \in R^i_{g_1}$ and $\forall v \in Y$ there exists $X_v \subseteq X$ such that $(v, X_v) \in R^i_{g_2}$ and $\bigcup_{v \in Y} X_v = X\}$.

– $R^i_{g_1 \cup g_2} = R^i_{g_1} \cup R^i_{g_2}$.

Iteration The semantics of Kleene star is defined with respect to a least fixed-point operator. We formalize this as follows: let $\cdot$ be a binary operator over $W \times 2^W$ which is defined as,

– $R_1 \cdot R_2 = \{(u, X) \mid \exists w_1, Y_1, \ldots, w_k, Y_k$ with $(u, \{w_1, \ldots, w_k\}) \in R_1, \forall j, (w_j, Y_j) \in R_2$ and $X = \bigcup_j Y_j\}$.

for all $R_1, R_2 \subseteq W \times 2^W$.

Given a $Z \subseteq W \times 2^W$, let F_Z be the operator over the domain $W \times 2^W$ defined as $F_Z(R) = R_\top \cup Z \cdot R$ where $R_\top = \{(u, \{u\}) \mid u \in W\}$. Observe that the operator $\cdot$ is monotonic in the following sense: if $R_1 \subseteq R_2$ then $R_0 \cdot R_1 \subseteq R_0 \cdot R_2$. This also implies that F_Z is monotonic for every $Z \subseteq W \times 2^W$. Thus by the Knaster-Tarski theorem we have that for every Z, the least fixed-point (LFP) of F_Z exists. $LFP(F_Z)$ can be computed as the limit of the following sequence of partial solutions: $R_0 = R_\top, R_{j+1} = F_Z(R_j)(= R_\top \cup Z \cdot R_i)$ and $R_\lambda = \cup_{\nu<\lambda} R_\nu$ for a limit ordinal λ.

– $R^i_{g^*} = LFP(F_{R^i_g})$.

Truth The truth of a formula $\alpha \in \mathcal{L}_c$ in a model M and a position s (denoted $M, s \models \alpha$) is defined as follows:

– $M, s \models p$ iff $p \in V(s)$.
– $M, s \models \neg\alpha$ iff $M, s \not\models \alpha$.
– $M, s \models \alpha_1 \vee \alpha_2$ iff $M, s \models \alpha_1$ or $M, s \models \alpha_2$.
– $M, s \models \langle g, i \rangle \alpha$ iff $\exists (s, X) \in R^i_g$ such that $\forall s' \in X$ we have $M, s' \models \alpha$.

A formula α is satisfiable if there exists a model M and a state s such that $M, s \models \alpha$.

5.5 Axiom system

We present an axiomatization of the valid formulas of the logic. Let $h^i_a = ((i, x), a, (j, y))$ and $h^{\bar{\imath}}_a = ((\bar{\imath}, x), a, (j, y))$. We can then define $\langle a \rangle \alpha$ with the standard modal logic interpretation as follows:

– $\langle a \rangle \alpha = (\mathbf{turn}_i \rightarrow \langle (h^i_a, \top), i \rangle \alpha) \wedge (\mathbf{turn}_{\bar{\imath}} \rightarrow \langle (h^{\bar{\imath}}_a, \top), \bar{\imath} \rangle \alpha)$.

For $h \in \mathbb{G}(Nodes)$, we use the notation $h^{\checkmark}$ to denote that the tree structure h is enabled. This is defined as follows:

– If $h = (i, x)$ then $h^{\checkmark} = \top$.
– If $h = ((i, x), a_1, t_{a_1}) + \ldots + ((i, x), a_k, t_{a_k})$ then
  - $h^{\checkmark} = \mathbf{turn}_i \wedge (\bigwedge_{a_j \in moves(s_0)} (\langle a_j \rangle \top \wedge [a_j] t^{\checkmark}_{a_j}))$.

The axiom schemes

(A1) Propositional axioms:
  (a) All the substitutional instances of tautologies of PC.
  (b) $\mathbf{turn}_i \leftrightarrow \neg\mathbf{turn}_{\bar{\imath}}$.

(A2) Axiom for single edge games:
  (a) $\langle a\rangle(\alpha_1 \vee \alpha_2) \leftrightarrow \langle a\rangle\alpha_1 \vee \langle a\rangle\alpha_2$.
  (b) $\langle a\rangle\mathbf{turn}_i \rightarrow [a]\mathbf{turn}_i$.

(A3) Dynamic logic axioms:
  (a) $\langle g_1 \cup g_2, i\rangle\alpha \leftrightarrow \langle g_1, i\rangle\alpha \vee \langle g_2, i\rangle\alpha$.
  (b) $\langle g_1; g_2, i\rangle\alpha \leftrightarrow \langle g_1, i\rangle\langle g_2, i\rangle\alpha$.
  (c) $\langle g^*, i\rangle\alpha \leftrightarrow \alpha \vee \langle g, i\rangle\langle g^*, i\rangle\alpha$.

(A4) $\langle (h, \beta), i\rangle\alpha \leftrightarrow h^{\checkmark} \wedge \downarrow_{(h,i,\beta,\alpha)}$.

where we define $\downarrow_{(h,i,\beta,\alpha)}$ as follow:

– if $h = (j, x)$ for $j \in N$ then $\downarrow_{(h,i,\beta,\alpha)} = \beta \wedge \alpha$.
– if $h = ((j, x), a_1, t_{a_1}) + \ldots + ((j, x), a_k, t_{a_k})$ with $A = \{a_1, \ldots, a_k\}$ then
  - $\downarrow_{(h,i,\beta,\alpha)} = \begin{cases} \bigvee_{a \in A} \langle a\rangle\langle (t_a, \beta), i\rangle\alpha \text{ if } j = i. \\ \bigwedge_{a \in A} [a]\langle (t_a, \beta), i\rangle\alpha \text{ if } j = \bar{\imath}. \end{cases}$

Inference rules

$$(MP)\,\frac{\alpha, \quad \alpha \rightarrow \beta}{\beta} \qquad (NG)\,\frac{\alpha}{[a]\alpha}$$

$$(IND)\,\frac{\langle g, i\rangle\alpha \rightarrow \alpha}{\langle g^*, i\rangle\alpha \rightarrow \alpha}$$

For a proof of completeness of this axiom system as well as decidability of the logic, see [15].

6 Dynamics of large games

We have merely touched on logical and finite memory structure in strategies in games. Hopefully, all the preceding discussion convinces the reader that granting first class citizenship to strategies is worthwhile and feasible. The question remains whether such an approach yields new and interesting insights. Below we mention some directions in which such a study might proceed, offering only a graphic outline of ideas and trends; a fully formulated theory awaits development. The main strand running through the discussion is that we consider temporally large games (that have episodic structure) and spatially large games (where the number of players is too large for rationality to be based on exhaustive intersubjectivity).

6.1 Strategy switching and stability

We have argued that resource limited players do not select complete strategies. Rather, they start initially with a set of possible strategies, knowledge about the game and other players' skills. As the game progresses, they compose/switch to devise new strategies. This can be specified in a syntax for strategy specification that crucially uses a construct for players to play the game with a strategy ν_1 up to some point and then switch to a strategy ν_2.

$$\Omega_i ::= \nu \in \Sigma_i \mid \mathit{Strat}_1 \cup \mathit{Strat}_2 \mid \mathit{Strat}_1 \cap \mathit{Strat}_2 \mid \mathit{Strat}_1{}^\frown \mathit{Strat}_2 \mid \mathit{Strat}_1 + \mathit{Strat}_2 \mid \psi ? \mathit{Strat}$$

Using the "test operator" $\psi?\mathit{Strat}$, a player checks whether an observable condition ψ holds and then decides on a strategy. We think of these conditions as past time formulas of a simple tense logic over an atomic set of observables.

In the atomic case, ν simply denotes a partial strategy. The intuitive meaning of the operators are given as:

- $\mathit{Strat}_1 \cup \mathit{Strat}_2$ means that the player plays according to the strategy Strat_1 or the strategy Strat_2.
- $\mathit{Strat}_1 \cap \mathit{Strat}_2$ means that if at a history $t \in T$, Strat_1 is defined then the player plays according to Strat_1; else if Strat_2 is defined at t then the player plays according to Strat_2. If both Strat_1 and Strat_2 are defined at t then the moves that Strat_1 and Strat_2 specify at t must be the same (we call such a pair Strat_1 and Strat_2, compatible).
- $\mathit{Strat}_1{}^\frown \mathit{Strat}_2$ means that the player plays according to the strategy Strat_1 and then after some history, switches to playing according to Strat_2. The position at which she makes the switch is not fixed in advance.
- $(\mathit{Strat}_1 + \mathit{Strat}_2)$ says that at every point, the player can choose to follow either Strat_1 or Strat_2.
- $\psi?\mathit{Strat}$ says at every history, the player tests if the property ψ holds of that history. If it does then she plays according to Strat.

The following lemma relates strategy specifications to finite state transducers, which are automata that output advice. Below note that *Strat* is a strategy specification, a syntactic object, and μ is a (functional) strategy, defined earlier to be a subtree of the tree unfolding of the game arena.

Lemma 6.1. *Given game arena $\mathcal{G}$, a player $i \in N$ and a strategy specification $Strat \in \Omega_i$, where all the atomic strategies mentioned in Strat are bounded memory, we can construct a transducer $\mathcal{A}_{Strat}$ such that for all $\mu \in \Omega^i$ we have $\mathcal{G}, \mu \models Strat$ iff $\mu \in Lang(\mathcal{A}_{Strat})$.*

Call a strategy *Strat* switch-free if it does not have any of the $\frown$ or the $+$ constructs.

Given a game arena $\mathcal{G}$ and strategy specifications of the players, we may ask whether there exists some subarena of $\mathcal{G}$ that the game settles down to if the players play according to their strategy specifications. (Note that a play being an infinite path in a finite graph, by settling down, we refer to the connected component that the play is eventually confined to.) This subarena is in some sense the equilibrium states of the game. It is also meaningful to ask if the game settles down to such an equilibrium subarena, then whether the strategy of a particular player attains stability with respect to switching.

Theorem 6.1. *Given a game arena $\mathcal{G} = (W, \rightarrow, w_0)$ a subarena R of $\mathcal{G}$ and strategy specifications $Strat_1, \ldots, Strat_n$ for players 1 to n, the following questions are decidable.*

- *Do all plays conforming to these specifications eventually settle down to R?*
- *Given strategy specifications $Strat_1, \ldots, Strat_n$ for players 1 to n, if all plays conforming to these specifications converge to R, does the strategy of player i become eventually stable with respect to switching?*

For a detailed study of strategy switching, see [33].

6.2 Issues in games with a large number of players

Game models of social situations typically involve large populations of players. However, common knowledge of rationality symmetrizes player behaviour and allows us to predict behaviour of any rational player. On the other hand, it is virtually impossible for each player to reason about the behaviour of every other player in such games, since a player may not even know how many players are in the game, let alone how they are likely to play.

What is the technical implication of number of players being large? In our view, in large games, the payoffs are usually dependent on the 'distribution' of the actions played by the players rather than the action profiles themselves. Moreover, in such games the payoffs are independent of the identities of the players.

An action distribution is a tuple $\mathbf{y} = (y_1, y_2, \ldots, y_{|A|})$ such that $y_i \geq 0,\ \forall i$ and $\sum_{i=1}^{|A|} y_i \leq n$. Let $\mathbf{Y}$ be the set of all action distributions. Given an action profile $\mathbf{a}$, we let $\mathbf{y}(\mathbf{a})$ be its corresponding action distribution, that is, $\mathbf{y}(\mathbf{a})(k)$ gives the number of players playing the kth action in A. Every player i has a rational valued function $f_i : \mathbf{Y} \rightarrow \mathbb{Q}$ which can be seen as the payoff of i for a particular distribution.

Why are such distributions interesting? They are used in many social situations. For instance, recently Singapore decided to make the entire city Wi-Fi enabled. How is it decided that a facility be provided as infrastructure? Typically such analysis involves determining when usage crosses a threshold. But then understanding why usage of one facility increases vastly, rather than another, despite the presence of several alternatives, is tricky. But this is what strategy selection is about. However, we are not as much bothered about strategy selection by an individual player but by a significant fraction of the population.

Similar situations occur in the management of the Internet. Policies for bandwidth allocation are not static. They are dynamic, based on studying both volumes of traffic and type of traffic. The popularity of an application like YouTube dramatically changes such usage, calling for changes in Internet policies. Predicting such future requirements is tricky, but much wanted by the engineers. Herd mentality and imitation are common in such situations.

In large games, payoffs are associated not with strategy profiles, but with type distributions. Suppose there are k strategies used in the population. Then the outcome is specified as a map $\mu : \Pi_k(n) \rightarrow P^k$, where $\Pi_k(n)$ is a set of distributions: k-tuples that sum up to n, and P is a payoff. Thus every player playing the j^{th} strategy gets the payoff given by the j^{th} component specified by μ for a given distribution. Typically there is usually a small number t of types such that $t < n$ where n is the number of players. Can one carry out all the analysis using only the t types and then lift the results to the entire game?

Why should such an analysis be possible? When we confine our attention to finite memory players, for n players, the strategy space is the n-fold product of these memory states. What we wish to do is to map this space into a t-fold product, whereby we wish to identify two players of the same type. We can show that in the case of deterministic transducers, such a blow-up is avoidable, since the product of a type with itself is then isomorphic to the type.

A population of 1000 players with only two types needs to be represented only by pairs of states and not 1000-tuples. But we need to determinize transducers, and that leads to exponential blow-up. So one might ask, when is the determinising procedure worthwhile? Suppose we have n players, t types, and p is the maximum size of the state space of any nondeterministic type finite state transducer. It turns out ([30]) that the construction is worthwhile when $n > 0.693 \cdot t \cdot \pi(p)$, where $\pi(p)$ is the number of primes below p. As we are talking about large games, the inequality above can be expected to hold.

In general, while we have spoken only of qualitative outcomes, and this is natural for a logical study, it makes sense to consider **quantitative** objectives as well, especially when outcomes are distribution determined. For infinite play, such outcomes may diverge, and we then need to consider limit-average payoffs (or other discounted payoffs). It is still possible to carry out the kind of analysis as we have discussed here, to show existence of equilibria in finite memory strategies, as for instance, in [30].

Neighbourhood structures In large games, it is convenient to think of players arranged in neighbourhoods. A player strategizes locally, observing behaviour and outcomes within her neighbourhood, but may switch to an adjacent neighbourhood.

As an example, consider vegetable sellers in India. In Indian towns, it is still possible to see vegetable sellers who carry vegetables in baskets or pushcarts and set up shop in some neighbourhood. The location of their 'shop' changes dynamically, based on the seller's perception of demand for vegetables in different neighbourhoods in the town, but also on who else is setting up shop near her, and on her perception of how well these (or other) sellers are doing. Indeed, when she buys a lot of vegetables in the wholesale market, the choice of her 'product mix' as well as her choice of location are determined by a complex rationale. While the prices she quotes do vary depending on the general market situation, the neighbourhoods where she sells also influence the prices significantly: she knows that in the poorer neighbourhoods, her buyers cannot afford to pay much. She can be thought of as a small player in a large game, one who is affected to some extent by play in the entire game, but whose strategising is local where such locality is itself dynamic.

In the same town, there are other, relatively better off vegetable sellers who have fixed shops. Their prices and product range are determined largely by wholesale market situation, and relatively unaffected by the presence of the itinerant vegetable sellers. If at all, they see themselves in competition only against other fixed-shop sellers. They can be seen as big players in a large game.

What is interesting in this scenario is the movement of a large number of itinerant vegetable sellers across the town, and the resultant increase and decrease in availability of specific vegetables as well as their prices. We can see the vegetable market as composed of dynamic neighbourhoods that expand and contract, and the dynamics of such a structure dictates, and is in turn dictated by the strategies of itinerant players.

When we model games with such neighbourhood structures, the central question to study is that of stability of game configurations. When can we guarantee that game dynamics leads to a configuration that does not change from then on, or oscillate between fixed configurations? Do finite memory strategies suffice? What kind of game theoretic tools are used for such analysis? [31] offers an instance, where a characterization is presented in terms of *potential games* [26]. However, the general question of what stable configurations are of interest and how to strategize to achieve them is of general interest, as well as obtaining bounds on when stability is attained.

Dynamic game forms Social situations often involve strategies that are generic, (almost) game-independent: threat and punishment; go with the winner / follow the leader; try to take the lead, and if you can't, follow a leader; imitate someone you think well of; and so on. They have some (limited) efficacy in many interaction situations. But when a significant proportion of players use such heuristics, it may affect game dynamics significantly.

In general, we can consider such dynamics as follows. An individual player has to make choices; making choices has a cost. Society provides choices, incurs cost to do so. Society revises choices and costs from time to time based on the history and prediction of the future. This effects individual strategies who switch between the available choices. Then the game arena is not static but changes dynamically.

We can then ask several questions based on eventual patterns dictated by the dynamics:

– Does the play finally settle down to some subset of the game?
– Can a player ensure certain objectives using a strategy that doesn't involve switching?
– Given a subarena, is a particular strategy *live*?
– Does an action profile eventually become part of the social infrastructure?
– Do the rules of the society and the behaviour of other players drive a particular player out of the game?

[32] offers a formal model in which such questions are posed and it is shown that these can be checked algorithmically. Therefore, it is possible to compare between game restriction rules in terms of their imposed social cost. For a player, if the game restriction rules are known and the type of the other players are known then she can compare between her strategy specifications.

The more general objective of such study is to explore the rationale of when and how should society intervene, and when such rationale is common knowledge among players, how they should strategize. In this sense, individual rationality and societal rationality are mutually recursive in each other, and the study of such interdependence offers an interesting challenge for logical models.

The imitation heuristic In a large population of players, where resources and computational abilities are asymmetrically distributed, it is natural to consider a population where the players are predominantly of two kinds: optimisers and imitators. Asymmetry in resources and abilities can then lead to different types of imitation and thus ensure that we do not end up with "herd behaviour". Mutual reasoning and strategising process between optimizers and imitators leads to interesting questions for game dynamics in these contexts.

Is imitation justified? We can say no since it does not achieve optimal in most cases. But we can also say yes, since it saves time, uses less resource and does not do much worse than optimal outcomes in most cases.

The rationality (or otherwise) of imitation has been studied (though perhaps not extensively) in game theory. In [29], games of unbounded duration on finite graphs are studied, where players may have overlapping objectives, and are divided into players who optimise and others who imitate. In this setting, it is shown that the following questions can be answered algorithmically:

– If the optimisers and the imitators play according to certain specifications, is a global outcome eventually attained?
– What sort of imitative behaviour (subtypes) eventually survive in the game?
– How worse-off are the imitators from an equilibrium outcome?

However, this is a preliminary study, and more sophisticated models would involve randomizing players as well as more nuanced player types in the population.

6.3 A research agenda

An important area that we have not touched on at all is that of games of imperfect information. This is especially important since *communication structure* in strategies is important and worthwhile for study. What to communicate and when it is strategic and when players have only a partial view of the game state and must communicate to learn more as well as coordinate to achieve desired goals, this dictates considerable structure in strategies. For an instance of such reasoning, see [40].

Below we list several questions that come up in the course of a search for strategy structure.

– We have discussed games with a fixed number of players (albeit unknown perhaps). How is strategizing affected when the set of players is *unbounded*, and hence potentially infinite? This is the case in games such as the Internet.
– We discussed neighbourhood structures, but in the model, we have merely replaced a flat structure on players by one which has one level depth. It is natural to consider a hierarchical structure of neighbourhoods, and a topological study would be more useful.
– We have talked only of player behaviour in games. A closely related question is one that keeps the strategy space fixed, but asks for *incentive mechanisms* that achieve desired outcomes. Mechanism design in the context of structured strategies is unclear.
– We have suggested that heuristics such as imitation are important in large games. It would be interesting to offer such analysis for a study of herd behaviour and runaway phenomena.
– At a foundational level, what we seek is an *algebraic theory* of strategies. What operators should be considered and how they interact requires a deeper mathematical study.
– We talked of game - strategy pairs, to show that they are dependent notions. A theory in which games and strategies are mutually recursive in the other is needed for offering foundations to such reasoning.
– Finite state transducers provide a natural *complexity measure* for strategies: the size of the minimal deterministic finite state machine that can play that strategy. Developing a nuanced complexity theory of strategies based on such notions is a definite need. This requires notions of *strategy reductions* that await further exploration.

– A most critical lacuna in our discussion has been the omission of *randomized* strategies. Logical theories that admit strategy structure as well as randomization are essential for applicability.

To conclude, singing praise of strategies is not only interesting in itself, but can also offer new questions for study to both game theorists and logicians.

References

1. Anderson, J.: How Can the Human Mind Occur in the Physical Universe? Oxford University Press, New York (NY) (2007)
2. van Benthem, J., Ghosh, S., Liu, F.: Modelling simultaneous games with dynamic logic. Knowledge, Rationality and Action 165, 247–268 (2008)
3. Berlekamp, E., Conway, J., Guy, R.: Winning Ways for Your Mathematical Plays, Volume 1. A.K. Peters, 2nd edn. (2001)
4. Blackburn, P., de Rijke, M., Venema, Y.: Modal Logic. CUP (2001)
5. Borgo, S.: Coalition in action logic. In: Veloso, M. (ed.) Proceedings of IJCAI'07. pp. 1822–1827 (2007)
6. Bouton, C.: Nim, a game with a complete mathematical theory. The Annals of Mathematics, 2nd Ser. 3(1/4), 35–39 (1902)
7. Chatterjee, K., Henzinger, T., Piterman, N.: Strategy logic. pp. 59–73. Springer-Verlag (2007)
8. Chellas, B.: Modal Logic: An Introduction. CUP (1980)
9. Fisher, R., Ury, W., Patton, B.: Getting to Yes: Negotiating Agreements Without Giving in. Penguin Books, Harmondsworth (1995)
10. Gale, D.: The game of hex and Brouwer fixed-point theorem. The American Mathematical Monthly 86, 818–827 (1979)
11. Gale, D., Stewart, F.: Infinite games with perfect information. In: Contributions to the Theory of Games, Volume 2, Annals of Mathematics Studies, vol. 28, pp. 245–266. Princeton University Press (1953)
12. Ghosh, S., Meijering, B.: On combining cognitive and formal modeling: a case study involving strategic reasoning. In: Proceedings of the workshop on Reasoning about other minds (RAOM 2011), CEUR Workshop Proceedings. vol. 751, pp. 79–92 (2011)
13. Ghosh, S., Meijering, B., Verbrugge, R.: Logic meets cognition: empirical reasoning in games. In: Proceedings of the 3rd International Workshop on Logics for Resource Bounded Agents (LRBA 2010), in 3rd Multi-Agent Logics, Languages, and Organisations Federated Workshops, MALLOW'10, CEUR Workshop Proceedings. vol. 627, pp. 15–34 (2010)
14. Ghosh, S., Ramanujam, R., Simon, S.: Playing extensive form games in parallel. In: Dix, J., others (eds.) Proceedings of the 11th International Workshop on Computational Logic in Multi-Agent Systems (CLIMA XI). LNCS, vol. 6245, pp. 153–170. Springer, Berlin (2010)
15. Ghosh, S., Ramanujam, R., Simon, S.: On strategy composition and game composition (2011), manuscript
16. Goldblatt, R.: Parallel action: Concurrent dynamic logic with independent modalities. Studia Logica 51, 551–578 (1992)
17. Harel, D., Kozen, D., Parikh, R.: Process logic: Expressiveness, decidability, completeness. Journal of Computer and System Sciences 25(2), 144–170 (1982)
18. Harel, D., Kozen, D., Tiuryn, J.: Dynamic Logic. The MIT Press (2000)
19. Hedden, T., Zhang, J.: What do you think I think you think? Strategic reasoning in matrix games. Cognition 85, 1–36 (2002)
20. Hoek, W.v.d., Jamroga, W., Wooldridge, M.: A logic for strategic reasoning. In: Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 05). pp. 157–164. ACM Inc, New York (2005)
21. Juvina, I., Taatgen, N.A.: Modeling control strategies in the n-back task. In: Proceedings of the 8th International Conference on Cognitive Modeling. Psychology Press, New York, NY (2007)

22. Kleene, S.: Representation of events in nerve nets and finite automata. In: Automata Studies, Annals of Mathematics Studies, vol. 34, pp. 3–41. Princeton University Press (1956)
23. Lovett, M.C.: A strategy-based interpretation of Stroop. Cognitive Science 29(3), 493–524 (2005)
24. Martin, D.: Borel Determinacy. Annals of Mathematics 102, 363–371 (1975)
25. Meijering, B., Maanen, L.v., Rijn, H.v., Verbrugge, R.: The facilitative effect of context on second-order social reasoning. In: Proceedings of the 32nd Annual Meeting of the Cognitive Science Society, Cognitive Science Society (2010)
26. Monderer, D., Shapley, L.: Potential games. Games and Economic Behaviour 14, 124–143 (1996)
27. Nash, J.: Equilibrium points in n-person games. Proceedings of the National Academy of Sciences 36, 89–93 (1950)
28. Parikh, R.: The logic of games and its applications. In: Selected papers of the international conference on "foundations of computation theory" on Topics in the theory of computation. pp. 111–139. Elsevier North-Holland, Inc., New York, NY, USA (1985)
29. Paul, S., Ramanujam, R.: Imitation in large games. In: Proceedings of the first international symposium on games, automata, logics and verification (GandALF). pp. 162–172. Electronic proceedings in theoretical computer science (2010)
30. Paul, S., Ramanujam, R.: Dynamic restriction of choices: Synthesis of societal rules. In: van Ditmarsch, H., Lang, J., Ju, S. (eds.) LORI. Lecture Notes in Computer Science, vol. 6953, pp. 28–50. Springer (2011)
31. Paul, S., Ramanujam, R.: Neighbourhood structure in large games. In: Proceedings of the 13th conference on the Theoretical Aspects of Rationality and Knowledge (TARK 2011). pp. 121–130 (2011)
32. Paul, S., Ramanujam, R., Simon, S.: Dynamic restriction of choices: A preliminary logical report. In: Proceedings of the 12th Conference on Theoretical Aspects of Rationality and Knowledge (TARK). pp. 218–226 (2009)
33. Paul, S., Ramanujam, R., Simon, S.: Stability under strategy switching. In: Ambos-Spies, K., Löwe, B., Merkle, W. (eds.) Proceedings of the 5th Conference on Computability in Europe (CiE). LNCS, vol. 5635, pp. 389–398 (2009)
34. Pauly, M.: Logics for Social Software. Ph.D. thesis, University of Amsterdam (2001)
35. Pinchinat, S.: A generic constructive solution for concurrent games with expressive constraints on strategies. pp. 253–267. Springer-Verlag (2007)
36. Polya, G.: How to Solve It. A New Aspect of Mathematical Method. Princeton University Press, Princeton, N.J. (1945)
37. Raiffa, H.: The Art and Science of Negotiation. Harvard University Press, Cambridge (MA) (1982)
38. Raiffa, H., Richardson, J., Metcalfe, D.: Negotiation Analysis: The Science and Art of Collaborative Decision Making. Belknap Press of Harvard Univ. Press, Cambridge (MA) (2002)
39. Ramanujam, R., Simon, S.: Structured strategies in games on graphs. In: Logic and Automata, Text in Logic and Games, vol. 2, pp. 553–566. AUP (2007)
40. Ramanujam, R., Simon, S.: A communication based model for games of imperfect information. In: Gastin, P., Laroussinie, F. (eds.) CONCUR. Lecture Notes in Computer Science, vol. 6269, pp. 509–523. Springer (2010)
41. Simon, S.: A logical study of strategies in games. Ph.D. thesis, The Institute of Mathematical Sciences, Chennai (August 2009)
42. Trapa, P., Novak, M.: Nash equilibria for an evolutionary language game. Journal of Mathematical Biology 41, 172–188 (2000)

43. Walther, D., Hoek, W.v.d., Wooldridge, M.: Alternating-time temporal logic with explicit strategies. In: Proceedings of XIth Conference (Theoretical Aspects of Rationality and Knowledge). pp. 269–278 (2007)
44. Zermelo, E.: Über eine Anwendung der Mengenlehre auf die Theorie des Schachspiels. In: Proceedings of the Fifth Congress of Mathematicians. pp. 501–504. CUP (1913)