Strategies: A logic - automata study
Lecture 3: Game logic and its descendants

Sujata Ghosh and R. Ramanujam

University of Groningen, The Netherlands

The Institute of Mathematical Sciences, Chennai, India

August 3, 2011

Verification games

Games that automata play:

- Two-player zero-sum game of perfect information played on (a typically infinite) graph arena.

Verification games

Games that automata play:

- Two-player zero-sum game of perfect information played on (a typically infinite) graph arena.
- Existence of winning strategies used to analyze possibility of automaton offering suitable response for all possible provocations from an uncertain environment.

Verification games

Games that automata play:

- Two-player zero-sum game of perfect information played on (a typically infinite) graph arena.
- Existence of winning strategies used to analyze possibility of automaton offering suitable response for all possible provocations from an uncertain environment.
- Important applications in control synthesis and thus in the design and verification of systems.

Verification games

Games that automata play:

- Two-player zero-sum game of perfect information played on (a typically infinite) graph arena.
- Existence of winning strategies used to analyze possibility of automaton offering suitable response for all possible provocations from an uncertain environment.
- Important applications in control synthesis and thus in the design and verification of systems.
- An emphasis on the size of *memory* needed.

Logic games

Games that logicians play:

- Associate a two-player zero-sum game of perfect information with the semantics of the logic.

Logic games

Games that logicians play:

- Associate a two-player zero-sum game of perfect information with the semantics of the logic.
- Logical notions like satisfiability are reduced to existence of winning strategies for one of the players.

Logic games

Games that logicians play:

- Associate a two-player zero-sum game of perfect information with the semantics of the logic.
- Logical notions like satisfiability are reduced to existence of winning strategies for one of the players.
- A tradition of use in model theory, for model construction, comparing models etc.

Economists' games

Games that economists play:

- Model a market or economic situation as a game, usually with a small number of players, with a small set of options.

Economists' games

Games that economists play:

- Model a market or economic situation as a game, usually with a small number of players, with a small set of options.
- Study equilibria to predict rational play; an emphasis on quantitative solutions.

Economists' games

Games that economists play:

- Model a market or economic situation as a game, usually with a small number of players, with a small set of options.
- Study equilibria to predict rational play; an emphasis on quantitative solutions.
- A tradition of use in market design, information economics and some political analysis.

Structure of strategies

- *Strategies are the unsung heroes of game theory.*
  – Johan van Benthem

Structure of strategies

- *Strategies are the unsung heroes of game theory.*
  – Johan van Benthem
- Perhaps combinatorial game theory is the only exception.

Structure of strategies

- *Strategies are the unsung heroes of game theory.*
  – Johan van Benthem
- Perhaps combinatorial game theory is the only exception.
- Certainly, this is true of the studies above.

Structure in games

We argued in Lecture 1 that strategizing by computationally limited agents is local and heuristic.

- This suggests compositional structure in strategies.

Structure in games

We argued in Lecture 1 that strategizing by computationally limited agents is local and heuristic.

- This suggests compositional structure in strategies.
- Something that logic is particularly good at.

Structure in games

We argued in Lecture 1 that strategizing by computationally limited agents is local and heuristic.

- This suggests compositional structure in strategies.
- Something that logic is particularly good at.
- And this is best done by looking for compositional structure in games. For this viewing games as programs is useful.

Structured programs

Let Σ be a set of *atomic* programs. More complex programs are built from these using program composition.

$$Pr ::= a \in \Sigma \mid \pi_1 + \pi_2 \mid \pi_1 ; \pi_2 \mid \pi^*$$

- This is the class of *regular* or finite-state programs.

Regular operations

Kleene (1953) showed that these operations suffice to capture all finite-state behaviours.

- $\pi_1 + \pi_2$ stands for choice; do either of the two.

Regular operations

Kleene (1953) showed that these operations suffice to capture all finite-state behaviours.

- $\pi_1 + \pi_2$ stands for choice; do either of the two.
- $\pi_1 ; \pi_2$ stands for sequential composition; do π_1 first and then do π_2.

Regular operations

Kleene (1953) showed that these operations suffice to capture all finite-state behaviours.

- $\pi_1 + \pi_2$ stands for choice; do either of the two.
- $\pi_1 ; \pi_2$ stands for sequential composition; do π_1 first and then do π_2.
- π^* stands for iteration; do π repeatedly, arbitrarily many times.

Regular operations

Kleene (1953) showed that these operations suffice to capture all finite-state behaviours.

- $\pi_1 + \pi_2$ stands for choice; do either of the two.
- $\pi_1 ; \pi_2$ stands for sequential composition; do π_1 first and then do π_2.
- π^* stands for iteration; do π repeatedly, arbitrarily many times.
- Assignment statements, input / output are all among atomic programs in Σ.

State transformers

Let S be a set of states. We can think of a program π as a map from S to 2^S: $\pi(s)$ denotes the set of states that the program may reach, when started operation from s.

State transformers

Let S be a set of states. We can think of a program π as a map from S to 2^S: $\pi(s)$ denotes 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 when started from s.

State transformers

Let S be a set of states. We can think of a program π as a map from S to 2^S: $\pi(s)$ denotes 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 when started from s.
- If we were to see X as a 'goal', a program then sounds very much like a strategy to achieve the goal.

Predicate transformers

If α is a property (defined on states), we can think of a program π as a set of pairs (α, β); if π is started in a state where α holds, when π terminates, the resulting state is one in which β holds.

Predicate transformers

If α is a property (defined on states), we can think of a program π as a set of pairs (α, β); if π is started in a state where α holds, when π terminates, the resulting state is one in which β holds.

- The properties are stated in a logical language. Let P be a set of atomic propositions.

Predicate transformers

If α is a property (defined on states), we can think of a program π as a set of pairs (α, β); if π is started in a state where α holds, when π terminates, the resulting state is one in which β holds.

- The properties are stated in a logical language. Let P be a set of atomic propositions.
- Formulas of the logic are built from P by closing under $\neg$ (negation), and $\vee$ ('or'), and the condition: if α is a formula, then $< \pi > \alpha$ is a formula as well.

Predicate transformers

If α is a property (defined on states), we can think of a program π as a set of pairs (α, β); if π is started in a state where α holds, when π terminates, the resulting state is one in which β holds.

- The properties are stated in a logical language. Let P be a set of atomic propositions.
- Formulas of the logic are built from P by closing under $\neg$ (negation), and $\vee$ ('or'), and the condition: if α is a formula, then $< \pi > \alpha$ is a formula as well.
- This is Propositional dynamic logic.

Nondeterministic programs

Consider the following (non-deterministic) program: let x, y be natural numbers.
(**if** $x \geq y$ **then** $y := y + 1$) or (**if** $x \leq y$ **then** $x := x + 1$)

Nondeterministic programs

Consider the following (non-deterministic) program: let x, y be natural numbers.
(**if** $x \geq y$ **then** $y := y + 1$) or (**if** $x \leq y$ **then** $x := x + 1$)

- When $x = y$, the program behaves non-deterministically: either x or y is incremented.

Nondeterministic programs

Consider the following (non-deterministic) program: let x, y be natural numbers.
(**if** $x \geq y$ **then** $y := y + 1$) or (**if** $x \leq y$ **then** $x := x + 1$)

- When $x = y$, the program behaves non-deterministically: either x or y is incremented.
- Such programs can be thought as *1-player games*: when $x = y$, the player (Nature) has a choice of which transition to do.

Programs as games

We can now give a program π a game theoretic interpretation:
$\pi : 2^S \rightarrow 2^S$ such that, for $X \subseteq S$,
$\pi(X) = Y$ denotes the following:

Programs as games

We can now give a program π a game theoretic interpretation:
$\pi : 2^S \rightarrow 2^S$ such that, for $X \subseteq S$,
$\pi(X) = Y$ denotes the following:

> *Y is the set of states starting from which Nature has a strategy to reach the set X of states.*

Two player games

Does the addition of a second player make any essential difference ?

- Consider a game $\pi = (a + b); (e + f)$ where player I chooses to do either an a or b, and then player II chooses to do e or f.

Two player games

Does the addition of a second player make any essential difference ?

- Consider a game $\pi = (a + b); (e + f)$ where player I chooses to do either an a or b, and then player II chooses to do e or f.
- We can think of it as a sequential composition of two one player games $(a + b)$ and $(e + f)$ with rôles of player and opponent "switched" in the two games.

Two player games

Does the addition of a second player make any essential difference ?

- Consider a game $\pi = (a + b); (e + f)$ where player I chooses to do either an a or b, and then player II chooses to do e or f.
- We can think of it as a sequential composition of two one player games $(a + b)$ and $(e + f)$ with rôles of player and opponent "switched" in the two games.
- This idea leads us to Propositional game logic, which is similar to program logic, but admitting a player and an opponent.

Game logic

Parikh 1983: Propositional game logic, studies how a player's "power" evolves in a two-player game.

Game logic

Parikh 1983: Propositional game logic, studies how a player's "power" evolves in a two-player game.
Let G_0 be a set of *atomic* games. More complex games are built from these using game composition.

Game logic

Parikh 1983: Propositional game logic, studies how a player's "power" evolves in a two-player game.
Let G_0 be a set of *atomic* games. More complex games are built from these using game composition.

$$G ::= g \in G_0 \mid \pi_1 + \pi_2 \mid \pi_1 ; \pi_2 \mid \pi^* \mid \pi^d$$

Game logic

Parikh 1983: Propositional game logic, studies how a player's "power" evolves in a two-player game.
Let G_0 be a set of *atomic* games. More complex games are built from these using game composition.

$$G ::= g \in G_0 \mid \pi_1 + \pi_2 \mid \pi_1 ; \pi_2 \mid \pi^* \mid \pi^d$$

π^d stands for the game obtained when the game π is played with the players switching rôles.

Game logic

π^d stands for the game obtained when the game π is played with the players switching rôles.

- $(a + b); (e + f)^d$ stands for the game where one player chooses between a and b and then the other player chooses between e and f.

Game logic

π^d stands for the game obtained when the game π is played with the players switching rôles.

- $(a + b); (e + f)^d$ stands for the game where one player chooses between a and b and then the other player chooses between e and f.
- The formulas of the logic are as defined before: $< \pi > \alpha$ now denotes that Player I has a strategy for playing game π and achieving α.

Formalizing logic games

Game logic formalizes evaluation games in logics.

- Two player zero-sum game associated with any classical logic.

Formalizing logic games

Game logic formalizes evaluation games in logics.

- Two player zero-sum game associated with any classical logic.
- A formula and a structure are given. Player I wants the formula to be true in the structure, II wants it to be false.

Formalizing logic games

Game logic formalizes evaluation games in logics.

- Two player zero-sum game associated with any classical logic.
- A formula and a structure are given. Player I wants the formula to be true in the structure, II wants it to be false.
- They go down recursively into the formula: I gets a turn when the connective is $\vee$ or $\exists$. Similarly, II moves for $\wedge$ and $\forall$. Negation is role reversal.

Formalizing logic games

Game logic formalizes evaluation games in logics.

- Two player zero-sum game associated with any classical logic.
- A formula and a structure are given. Player I wants the formula to be true in the structure, II wants it to be false.
- They go down recursively into the formula: I gets a turn when the connective is $\vee$ or $\exists$. Similarly, II moves for $\wedge$ and $\forall$. Negation is role reversal.
- These are the game logic operators. Fixed point constructors in the logic are associated with game iteration.

Game logic [Parikh 83]

A logic to reason about determined two person zero sum games.

Syntax

- $\Phi := p \mid \neg\alpha \mid \alpha_1 \vee \alpha_2 \mid \langle\gamma\rangle\alpha$.

Game logic [Parikh 83]

A logic to reason about determined two person zero sum games.

Syntax

- $\Phi := p \mid \neg\alpha \mid \alpha_1 \vee \alpha_2 \mid \langle\gamma\rangle\alpha$.
- $\Gamma := g \in \Gamma_0 \mid \gamma_1 ; \gamma_2 \mid \gamma_1 \cup \gamma_2 \mid \gamma^* \mid \gamma^d$.
  - Sequential composition - $\gamma_1 ; \gamma_2$.
  - Choice - $\gamma_1 \cup \gamma_2$.
  - Iteration - γ^*.
  - Dual - γ^d.

Game logic

Syntax

- $\Phi := p \mid \neg\alpha \mid \alpha_1 \vee \alpha_2 \mid \langle\gamma\rangle\alpha$.
- $\Gamma := g \in \Gamma_0 \mid \gamma_1;\gamma_2 \mid \gamma_1 \cup \gamma_2 \mid \gamma^* \mid \gamma^d$.

Interpretation for games

- Final outcomes which players can enforce.

Game logic

Syntax

- $\Phi := p \mid \neg\alpha \mid \alpha_1 \vee \alpha_2 \mid \langle\gamma\rangle\alpha$.
- $\Gamma := g \in \Gamma_0 \mid \gamma_1;\gamma_2 \mid \gamma_1 \cup \gamma_2 \mid \gamma^* \mid \gamma^d$.

Interpretation for games

- Final outcomes which players can enforce.
- Set of states S.

Game logic

Syntax

- $\Phi := p \mid \neg\alpha \mid \alpha_1 \vee \alpha_2 \mid \langle\gamma\rangle\alpha$.
- $\Gamma := g \in \Gamma_0 \mid \gamma_1;\gamma_2 \mid \gamma_1 \cup \gamma_2 \mid \gamma^* \mid \gamma^d$.

Interpretation for games

- Final outcomes which players can enforce.
- Set of states S.
- Effectivity relation - $E_g \subseteq S \times 2^S$
  - $(s, X) \in E_g$ iff starting at s, in game g, player 1 can enforce the outcome to be in X.

Game logic

- $\Phi := p \mid \neg\alpha \mid \alpha_1 \vee \alpha_2 \mid \langle\gamma\rangle\alpha$.
- $\Gamma := g \in \Gamma_0 \mid \gamma_1;\gamma_2 \mid \gamma_1 \cup \gamma_2 \mid \gamma^d \mid \gamma^*$.

Model $M = (S, \{E_g \mid g \in \Gamma_0\}, V)$.

Neighbourhood semantics

- $M, s \models \langle\gamma\rangle\alpha$ iff $\exists(s, X) \in E_\gamma$ such that $X \subseteq \{s' \mid M, s' \models \alpha\}$.

Game logic

- $\Phi := p \mid \neg\alpha \mid \alpha_1 \vee \alpha_2 \mid \langle\gamma\rangle\alpha$.
- $\Gamma := g \in \Gamma_0 \mid \gamma_1;\gamma_2 \mid \gamma_1 \cup \gamma_2 \mid \gamma^d \mid \gamma^*$.

Model $M = (S, \{E_g \mid g \in \Gamma_0\}, V)$.

Neighbourhood semantics

- $M, s \models \langle\gamma\rangle\alpha$ iff $\exists(s, X) \in E_\gamma$ such that $X \subseteq \{s' \mid M, s' \models \alpha\}$.
- There exists a strategy in game γ to ensure α.

Determined games

We talk about two person zero sum games of perfect information in this logic.

- The two players cannot have winning strategies for complementary winning positions, thus $\neg(<\pi>\alpha \wedge <\pi^{d}>\neg\alpha)$ is a valid formula, for every game π.

Determined games

We talk about two person zero sum games of perfect information in this logic.

- The two players cannot have winning strategies for complementary winning positions, thus $\neg(<\pi>\alpha\wedge<\pi^{d}>\neg\alpha)$ is a valid formula, for every game π.
- All games are determined, and hence $(<\pi>\alpha\vee<\pi^{d}>\neg\alpha)$ is valid.

Axiom system

- All the substitutional instances of tautologies of PC.
- $\langle g_1 \cup g_2 \rangle \alpha \equiv \langle g_1 \rangle \alpha \vee \langle g_2 \rangle \alpha$.
- $\langle g_1 ; g_2 \rangle \alpha \equiv \langle g_1 \rangle \langle g_2 \rangle \alpha$.
- $\langle g^* \rangle \alpha \equiv \alpha \vee \langle g \rangle \langle g^* \rangle \alpha$.
- $\langle g^d \rangle \alpha \equiv \neg \langle g \rangle \neg \alpha$.

Inference rules

$$(MP)\ \frac{\alpha, \quad \alpha \supset \beta}{\beta} \qquad (NG)\ \frac{\alpha \supset \beta}{\langle g \rangle \alpha \supset \langle g \rangle \beta}$$

$$(IND)\ \frac{\langle g \rangle \alpha \supset \alpha}{\langle g^* \rangle \alpha \supset \alpha}$$

Complete ?

The axiom system is easily seen to be sound.

- The system without the duality axiom is easily seen to be complete for the dual-free fragment of the logic.

Complete ?

The axiom system is easily seen to be sound.

- The system without the duality axiom is easily seen to be complete for the dual-free fragment of the logic.
- In 1983, Parikh conjectured that the system presented is indeed complete for game logic.

Complete ?

The axiom system is easily seen to be sound.

- The system without the duality axiom is easily seen to be complete for the dual-free fragment of the logic.
- In 1983, Parikh conjectured that the system presented is indeed complete for game logic.
- This remains an interesting open problem.

Technical results

- The satisfiability problem for the logic above is *EXPTIME*-complete.
- This is the same as that for propositional dynamic logic.

Technical results

- The satisfiability problem for the logic above is *EXPTIME*-complete.
- This is the same as that for propositional dynamic logic.
- Model checking game logic is equivalent to the same problem for the modal μ-calculus.

Technical results

- The satisfiability problem for the logic above is *EXPTIME*-complete.
- This is the same as that for propositional dynamic logic.
- Model checking game logic is equivalent to the same problem for the modal μ-calculus.
- Complexity of model checking is in $NP \cap co - NP$: a major open problem asks if it is in P.

Remarks on game logic

Game transitions are effectivity functions, or player's powers; hence naturally satisfy monotonicity.

- So intended models for GL are higher order transition systems, or neighbourhood models.

Remarks on game logic

Game transitions are effectivity functions, or player's powers; hence naturally satisfy monotonicity.

- So intended models for GL are higher order transition systems, or neighbourhood models.
- In classical game theory, notion of utility and action sets are primitive. GL is detatched from these notions.

Remarks on game logic

Game transitions are effectivity functions, or player's powers; hence naturally satisfy monotonicity.

- So intended models for GL are higher order transition systems, or neighbourhood models.
- In classical game theory, notion of utility and action sets are primitive. GL is detatched from these notions.
- Game expressions define rules for constructing extensive game forms over internal positions, while atomic game forms and utilities are provided in terms of world states.

Remarks on game logic

Game transitions are effectivity functions, or player's powers; hence naturally satisfy monotonicity.

- So intended models for GL are higher order transition systems, or neighbourhood models.
- In classical game theory, notion of utility and action sets are primitive. GL is detatched from these notions.
- Game expressions define rules for constructing extensive game forms over internal positions, while atomic game forms and utilities are provided in terms of world states.
- This mix of game theoretical concepts and computer science leads to a non-classical notion of great potential.

Relation to program logics

GL is not only more expressive than PDL but also Δ-PDL.

- We can define a notion of *bisimulation* over game models and show that formulas of game logic are invariant under bisimulation.

Relation to program logics

GL is not only more expressive than PDL but also Δ-PDL.

- We can define a notion of *bisimulation* over game models and show that formulas of game logic are invariant under bisimulation.
- When we consider the natural notion of *modal alternation depth* of a formula, we find that the alternation hierarchy for game logic formulas does not collapse.

Relation to program logics

GL is not only more expressive than PDL but also Δ-PDL.

- We can define a notion of *bisimulation* over game models and show that formulas of game logic are invariant under bisimulation.
- When we consider the natural notion of *modal alternation depth* of a formula, we find that the alternation hierarchy for game logic formulas does not collapse.
- There is a translation of game logic into propositional μ-calculus with 2 variables (and 1 variable does not suffice).

Translation

Every GL formula can be translated into an equivalent one in the propositional μ-calculus with two fixed point variables.

- Three mappings ^{X}, ^{Y} and ^{F} are defined.

Translation

Every GL formula can be translated into an equivalent one in the propositional μ-calculus with two fixed point variables.

- Three mappings ^{X}, ^{Y} and ^{F} are defined.
- ^{X}, ^{Y} translate game expressions into formulas with one variable X and Y respectively.

Translation

Every GL formula can be translated into an equivalent one in the propositional μ-calculus with two fixed point variables.

- Three mappings ^{X}, ^{Y} and ^{F} are defined.
- ^{X}, ^{Y} translate game expressions into formulas with one variable X and Y respectively.
- ^{F} is the propositional translation.

The map ^{X}

- $g^X = \Diamond X$.
- $(g_1 + g_2)^X = g_1^X \vee g_2^X$.
- $(g_1; g_2)^X = g_1^X[X := g_2^X]$.
- $(g^d)^X = \neg g^X[X := \neg X]$.
- $(g^*)^X = \mu Y.(X \vee g^Y)$.

The map ^{Y}

- $g^Y = \Diamond Y$.
- $(g_1 + g_2)^Y = g_1^Y \vee g_2^Y$.
- $(g_1; g_2)^Y = g_1^Y[Y := g_2^Y]$.
- $(g^d)^Y = \neg g^Y[Y := \neg Y]$.
- $(g^*)^Y = \mu X.(Y \vee g^X)$.

The map ^{F}

- $p^F = p$.
- $(\neg\phi)^F = \neg\phi^F$.
- $(\phi_1 \vee \phi_2)^F = \phi_1^F \vee \phi_2^F$.
- $(< g > \phi)^F = g^X[X := \phi^F]$.

The map ^{F}

- $p^F = p$.
- $(\neg\phi)^F = \neg\phi^F$.
- $(\phi_1 \vee \phi_2)^F = \phi_1^F \vee \phi_2^F$.
- $(< g > \phi)^F = g^X[X := \phi^F]$.

We can then show that ϕ and ϕ^F are equivalent.

Departures

Game logic provides a point of departure for the study of logic and games.

- Non-determined games.

Departures

Game logic provides a point of departure for the study of logic and games.

- Non-determined games.
- Multi-player games.

Departures

Game logic provides a point of departure for the study of logic and games.

- Non-determined games.
- Multi-player games.
- Overlapping objectives.

Departures

Game logic provides a point of departure for the study of logic and games.

- Non-determined games.
- Multi-player games.
- Overlapping objectives.
- More interesting game operators.

Explicit strategies

Players' strategic response need to take into account:

- Observable behaviour of the other players.

Explicit strategies

Players' strategic response need to take into account:

- Observable behaviour of the other players.
- Compositional structure of the game.

At the logical level:

- Game composition and structured strategies are *not* independent entities.

Explicit strategies

Players' strategic response need to take into account:

- Observable behaviour of the other players.
- Compositional structure of the game.

At the logical level:

- Game composition and structured strategies are *not* independent entities.
- Games and strategies need to be composed together.

N-person game logic

Game logic: $\langle\gamma\rangle\alpha$

N-person game logic

Game logic: $\langle\gamma\rangle\alpha$

N-person game logic: $\langle\gamma, i\rangle\alpha$

Syntax

- $\Phi := p \mid \neg\alpha \mid \alpha_1 \vee \alpha_2 \mid \langle\gamma, i\rangle\alpha$.
- $\Gamma := g \in \Gamma_0 \mid \gamma_1;\gamma_2 \mid \gamma_1 \cup \gamma_2 \mid \gamma^*$

N-person game logic

Game logic: $\langle\gamma\rangle\alpha$

N-person game logic: $\langle\gamma, i\rangle\alpha$

Syntax

- $\Phi := p \mid \neg\alpha \mid \alpha_1 \vee \alpha_2 \mid \langle\gamma, i\rangle\alpha$.
- $\Gamma := g \in \Gamma_0 \mid \gamma_1; \gamma_2 \mid \gamma_1 \cup \gamma_2 \mid \gamma^*$

Semantics

$\mathcal{M} = (S, \{\rho_g^i \mid g \in \Gamma_0\}, V)$, where $\rho_g^i \subseteq S \times \mathcal{P}(S)$ is monotonic.

What are the properties?

What are the properties?

- How do we define the semantics?

What are the properties?

- How do we define the semantics?
- What is the truth definition of $\langle \gamma_1 \cup \gamma_2, i \rangle \varphi$?

What are the properties?

- How do we define the semantics?
- What is the truth definition of $\langle \gamma_1 \cup \gamma_2, i \rangle \varphi$?
- What kind of validities does this logic have?

What are the properties?

- How do we define the semantics?
- What is the truth definition of $\langle \gamma_1 \cup \gamma_2, i \rangle \varphi$?
- What kind of validities does this logic have?
- Is $\neg(\langle \gamma, i \rangle \varphi \wedge \langle \gamma, j \rangle \neg \varphi)$ valid?

Any other reasonable operator for combining games?

- Choice, Sequential composition, Iteration

Any other reasonable operator for combining games?

- Choice, Sequential composition, Iteration
- We would now have a look at parallel composition of games.

Any other reasonable operator for combining games?

- Choice, Sequential composition, Iteration
- We would now have a look at parallel composition of games.
- We consider two ways of looking at it : Intersection and interleaving
- Intersection: neighbourhood models
- Interleaving: tree models

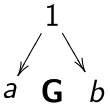
Simultaneous games

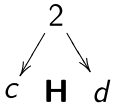
- Game models: $\mathcal{M} = (S, \{\rho_g^i \mid g \in \Gamma\}, V)$, where $\rho_g^i \subseteq S \times \mathcal{P}(S)$ is monotonic.
- Simultaneous game models: $\mathcal{M} = (S, \{\rho_g^i \mid g \in \Gamma\}, V)$, where $\rho_g^i \subseteq S \times \mathcal{P}(\mathcal{P}(S))$ is monotonic.
  The set lifting is considered to differentiate between union, intersection and parallel composition of games.

A toy example

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

A toy example





1's ability : $\{\{a\}\}, \{\{b\}\}$.
2' rability : $\{\{a\}, \{b\}\}$.

1's ability : $\{\{c\}, \{d\}\}$.
2's ability : $\{\{c\}\}, \{\{d\}\}$.

A toy example

1's ability : {{a, c}, {a, d}}, {{b, c}, {b, d}}.
2's ability : {{a, c}, {b, c}}, {{a, d}, {b, d}}.

Each outcome state is a set read ‘conjunctively’, but players have choices leading to sets of these read ‘disjunctively’ as in case of game logic.

Simultaneous two-player game logic

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

Simultaneous game model: $\mathcal{M} = (S, \{\rho^i_g \mid g \in \Gamma\}, V)$

$\mathcal{M}, s \models \langle \gamma, i \rangle \phi$ iff there exists $X : s\rho^i_\gamma X$ and $\forall x \in \cup X : \mathcal{M}, x \models \varphi$

Composite games

$$s\rho^1_{G\cup G'}X \quad \text{iff} \quad s\rho^1_G X \text{ or } s\rho^1_{G'}X.$$

$$s\rho^2_{G\cup G'}X \quad \text{iff} \quad s\rho^2_G X \text{ and } s\rho^2_{G'}X.$$

$$s\rho^1_{G^d}X \quad \text{iff} \quad s\rho^2_G X.$$

$$s\rho^2_{G^d}X \quad \text{iff} \quad s\rho^1_G X.$$

$$s\rho^i_{G;G'}X \quad \text{iff} \quad \exists U : s\rho^i_G U \text{ and for each } u \in \bigcup U,\ u\rho^i_{G'}X.$$

$$s\rho^i_{G\times G'}X \quad \text{iff} \quad \exists T, \exists W : s\rho^i_G T \text{ and } s\rho^i_{G'}W \text{ and } X = \{t \cup w : t \in T \text{ and } w \in W\}.$$

Complete axiomatization

a) all propositional tautologies and inference rules

b) if $\vdash \phi \rightarrow \psi$ then $\vdash \langle \gamma, i \rangle \phi \rightarrow \langle \gamma, i \rangle \psi$

c) reduction axioms:

$$\langle \alpha \cup \beta, 1 \rangle \phi \leftrightarrow \langle \alpha, 1 \rangle \phi \vee \langle \beta, 1 \rangle \phi$$
$$\langle \alpha \cup \beta, 2 \rangle \phi \leftrightarrow \langle \alpha, 2 \rangle \phi \wedge \langle \beta, 2 \rangle \phi$$
$$\langle \gamma^d, 1 \rangle \phi \leftrightarrow \langle \gamma, 2 \rangle \phi$$
$$\langle \gamma^d, 2 \rangle \phi \leftrightarrow \langle \gamma, 1 \rangle \phi$$
$$\langle \alpha; \beta, i \rangle \phi \leftrightarrow \langle \alpha, i \rangle \langle \beta, i \rangle \phi$$
$$\langle \alpha \times \beta, i \rangle \phi \leftrightarrow \langle \alpha, i \rangle \phi \wedge \langle \beta, i \rangle \phi$$

Complete axiomatization

a) all propositional tautologies and inference rules

b) if $\vdash \phi \rightarrow \psi$ then $\vdash \langle \gamma, i \rangle \phi \rightarrow \langle \gamma, i \rangle \psi$

c) reduction axioms:

$$\langle \alpha \cup \beta, 1 \rangle \phi \leftrightarrow \langle \alpha, 1 \rangle \phi \vee \langle \beta, 1 \rangle \phi$$
$$\langle \alpha \cup \beta, 2 \rangle \phi \leftrightarrow \langle \alpha, 2 \rangle \phi \wedge \langle \beta, 2 \rangle \phi$$
$$\langle \gamma^d, 1 \rangle \phi \leftrightarrow \langle \gamma, 2 \rangle \phi$$
$$\langle \gamma^d, 2 \rangle \phi \leftrightarrow \langle \gamma, 1 \rangle \phi$$
$$\langle \alpha; \beta, i \rangle \phi \leftrightarrow \langle \alpha, i \rangle \langle \beta, i \rangle \phi$$
$$\langle \alpha \times \beta, i \rangle \phi \leftrightarrow \langle \alpha, i \rangle \phi \wedge \langle \beta, i \rangle \phi$$

Consider $\langle \alpha \times \alpha, i \rangle \phi$!

Complete axiomatization

a) all propositional tautologies and inference rules

b) if $\vdash \phi \to \psi$ then $\vdash \langle \gamma, i \rangle \phi \to \langle \gamma, i \rangle \psi$

c) reduction axioms:

$$
\begin{aligned}
&\langle \alpha \cup \beta, 1 \rangle \phi \leftrightarrow \langle \alpha, 1 \rangle \phi \vee \langle \beta, 1 \rangle \phi \\
&\langle \alpha \cup \beta, 2 \rangle \phi \leftrightarrow \langle \alpha, 2 \rangle \phi \wedge \langle \beta, 2 \rangle \phi \\
&\langle \gamma^d, 1 \rangle \phi \leftrightarrow \langle \gamma, 2 \rangle \phi \\
&\langle \gamma^d, 2 \rangle \phi \leftrightarrow \langle \gamma, 1 \rangle \phi \\
&\langle \alpha; \beta, i \rangle \phi \leftrightarrow \langle \alpha, i \rangle \langle \beta, i \rangle \phi \\
&\langle \alpha \times \beta, i \rangle \phi \leftrightarrow \langle \alpha, i \rangle \phi \wedge \langle \beta, i \rangle \phi
\end{aligned}
$$

Consider $\langle \alpha \times \alpha, i \rangle \phi$! Composing the same games will give us back a single game, which do not agree with the spirit of what we were talking about earlier: Having games played in parallel, and copying strategies from one to the other.

Interleaving games

The general focus is on:

- an interleaving operator in the context of extensive form game trees looking into the structure of the games;
- information transfer from one game to another made possible by some common player enacting in all the games concerned;
- strategizing based on such information transfer.

This attempt

A formal study on the algebra of game composition: choice, iteration, sequential and parallel composition.

- A dynamic logic for game expressions extended with parallel composition.
- Extensive form games embedded in Kripke structures.
- Focus on interleavings of moves of players in the tree structure.

Composing game trees

Atomic games: Extensive form games

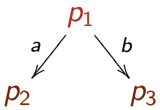
- Finite tree - nodes represent game positions labelled with players.
- Edge relation - specifies the moves which are enabled at a particular position.

Composite games:

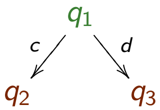
- choice $(g_1 \cup g_2)$;
- sequential composition $(g_1; g_2)$;
- iteration (g^*);
- parallel composition $(g_1 \| g_2)$.

Sequential composition

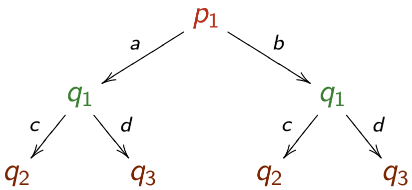
Sequential composition



T_1



T_2



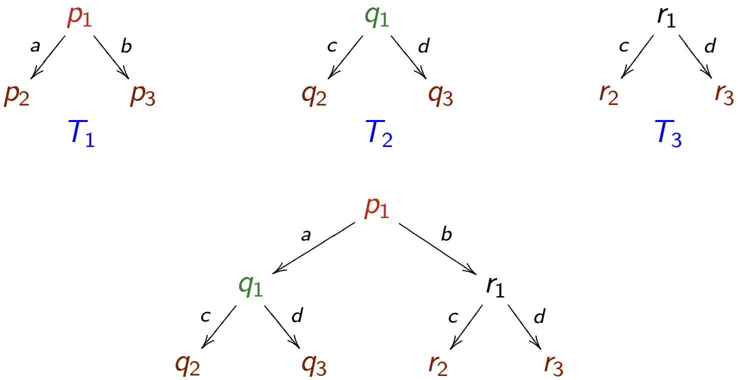
$T_1; T_2$

Sequential composition

$$
\begin{tikzcd}[column sep=small]
 & p_1 \arrow[dl, "a"'] \arrow[dr, "b"] & \\
p_2 & & p_3
\end{tikzcd}
\qquad
\begin{tikzcd}[column sep=small]
 & q_1 \arrow[dl, "c"'] \arrow[dr, "d"] & \\
q_2 & & q_3
\end{tikzcd}
\qquad
\begin{tikzcd}[column sep=small]
 & r_1 \arrow[dl, "c"'] \arrow[dr, "d"] & \\
r_2 & & r_3
\end{tikzcd}
$$

T_1 $\qquad$ T_2 $\qquad$ T_3

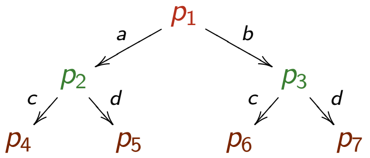
Sequential composition



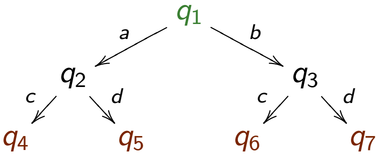
$$T \in T_1; \{T_2, T_3\}$$

Parallel composition

Same game; different players.

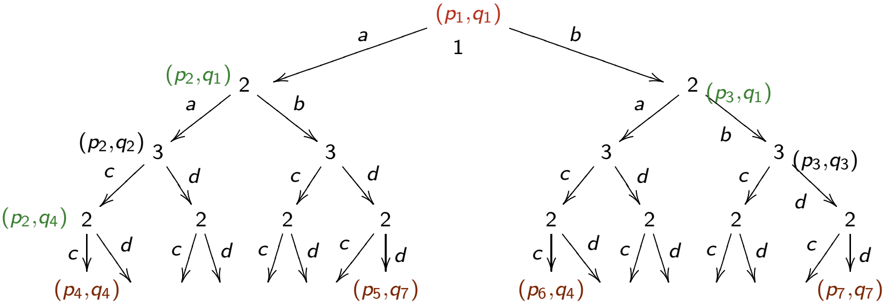


T_4



T_5

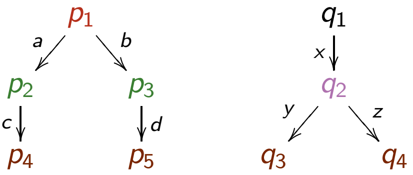
Parallel composition



$T \in T_4 \| T_5$

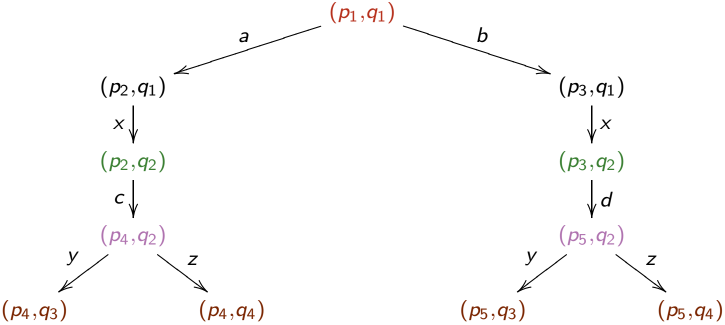
Parallel composition

Different games; different players.



T_6 T_7

Parallel composition



$$T \in T_6 \| T_7$$

The logic

Syntax

- $\Phi := p \in P \mid \neg\alpha \mid \alpha_1 \vee \alpha_2 \mid \langle g, i \rangle \alpha$.
- $\Gamma := h \in \mathbb{H} \mid g_1 ; g_2 \mid g_1 \cup g_2 \mid g_1 \| g_2$.

The logic

Syntax

- $\Phi := p \in P \mid \neg\alpha \mid \alpha_1 \vee \alpha_2 \mid \langle g, i \rangle \alpha$.
- $\Gamma := h \in \mathbb{H} \mid g_1 ; g_2 \mid g_1 \cup g_2 \mid g_1 \| g_2$.

Tree semantics

- $M, w \models \langle g, i \rangle \alpha$ iff there exists $X \in R_g^i$ such that X constitutes a valid tree, $root(X) = w$ and for all $v \in frontier(X)$, $M, v \models \alpha$. .

At a state w, a tree X in the "tree language" associated with g is enabled at w, and player i has a strategy (subtree) in it to ensure α.

Tree semantics

To define the semantic relation, we need to fix:

- Tree language associated with the game g.

Tree semantics

To define the semantic relation, we need to fix:

- Tree language associated with the game g.

Atomic games: Extensive form games

- Finite tree - nodes represent game positions labelled with players.
- Edge relation - specifies the moves which are enabled at a particular position.

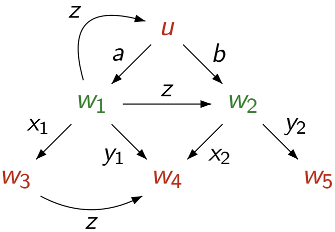
Model

- Model - Kripke structure.
  - A finite set of states W.
  - Labelled edge relation $\longrightarrow \subseteq W \times \Sigma \times W$.
  - Valuation function $V : W \to 2^P$.
  - Player labelling function $\lambda : W \to N$.

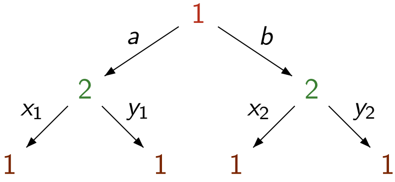
Model

- Model - Kripke structure.
  - A finite set of states W.
  - Labelled edge relation $\longrightarrow \subseteq W \times \Sigma \times W$.
  - Valuation function $V : W \rightarrow 2^P$.
  - Player labelling function $\lambda : W \rightarrow N$.
- Tree language for atomic games h.
  - if h is enabled at w, $R_h^i(w) \subseteq 2^{(W \times W)^*}$ encodes the set of all available strategies for player i in the game h enabled at w.
  - $R_h^i = \bigcup_{w \in W} R_h^i(w)$.

Game enabled

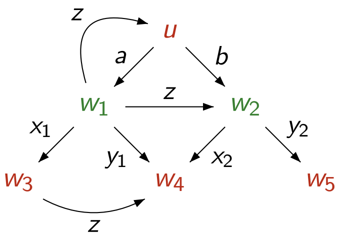


Model - M

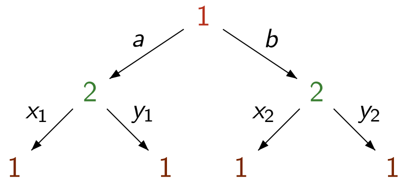


Game - h

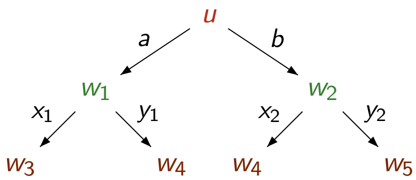
Game enabled



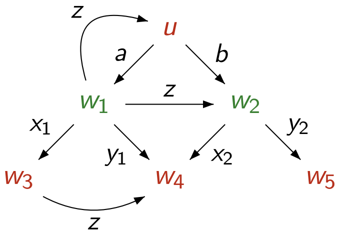
Model - M



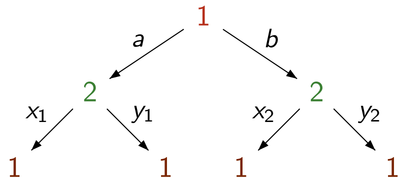
Game - h



Game enabled

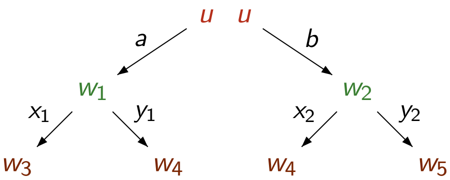


Model - M



Game - h

$$\{(u, w_1)(w_1, w_3), (u, w_1)(w_1, w_4)\},$$
$$\{(u, w_2)(w_2, w_4), (u, w_2)(w_2, w_5)\} \in R_h^1$$



Valid tree

Legal sequence ϱ: $(u, w)(v, x)$ in ϱ implies $w = v$.

$X \in 2^{(W \times W)^*}$ is a valid tree if:

- for all sequence $\varrho \in X$, ϱ is legal;
- X is prefix closed.

Valid tree

Legal sequence ϱ: $(u, w)(v, x)$ in ϱ implies $w = v$.

$X \in 2^{(W \times W)^*}$ is a valid tree if:

- for all sequence $\varrho \in X$, ϱ is legal;
- X is prefix closed.

Root and frontier

In a valid tree X,

- for all $\varrho, \varrho' \in X$, $\mathit{first}(\varrho)[1] = \mathit{first}(\varrho')[1] = \mathit{root}(X)$;
- $\mathit{frontier}(X) = \{\mathit{last}(\varrho)[2] \mid \varrho \in X\}$.

Composite game relations

- $X \in R^i_h \Longrightarrow$ tree structure $\mathfrak{T}(X)$
- finite game tree $\mathfrak{T} \Longrightarrow \mathfrak{f}(\mathfrak{T}) \in 2^{(W \times W)^*}$

Interpretation

Composite game relations

- $X \in R^i_h \Longrightarrow$ tree structure $\mathfrak{T}(X)$
- finite game tree $\mathfrak{T} \Longrightarrow \mathfrak{f}(\mathfrak{T}) \in 2^{(W \times W)^*}$

Interpretation

- $R^i_{g_1 \cup g_2} = R^i_{g_1} \cup R^i_{g_2}$;

Composite game relations

- $X \in R^i_h \Longrightarrow$ tree structure $\mathfrak{T}(X)$
- finite game tree $\mathfrak{T} \Longrightarrow \mathfrak{f}(\mathfrak{T}) \in 2^{(W \times W)^*}$

Interpretation

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

Composite game relations

- $X \in R^i_h \Longrightarrow$ tree structure $\mathfrak{T}(X)$
- finite game tree $\mathfrak{T} \Longrightarrow \mathfrak{f}(\mathfrak{T}) \in 2^{(W \times W)^*}$

Interpretation

- $R^i_{g_1 \cup g_2} = R^i_{g_1} \cup R^i_{g_2}$;
- $R^i_{g_1 ; g_2} = \{\mathfrak{f}(\mathfrak{T}(X); \mathcal{T}) \mid X \in R^i_{g_1}$ 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 $X_2 \in R^i_{g_2}\}$.

A satisfiable formula

Consider the game trees T_4 and T_5.

A satisfiable formula

Consider the game trees T_4 and T_5.

Suppose $M, w \models \bigwedge_{a_j \in \{a,b\}} (\langle a_j \rangle True \wedge$
$[a_j](\wedge_{a_j \in \{a,b\}} \langle a_j \rangle (\bigwedge_{a_j \in \{c,d\}} (\langle a_j \rangle True \wedge [a_j](\wedge_{a_j \in \{c,d\}} \langle a_j \rangle True));$

and, $M, w \models T_4^{\checkmark}$.

A satisfiable formula

Consider the game trees T_4 and T_5.

Suppose $M, w \models \bigwedge_{a_j \in \{a,b\}}(\langle a_j \rangle True \wedge$
$[a_j](\wedge_{a_j \in \{a,b\}} \langle a_j \rangle (\bigwedge_{a_j \in \{c,d\}}(\langle a_j \rangle True \wedge [a_j](\wedge_{a_j \in \{c,d\}} \langle a_j \rangle True));$
and, $M, w \models T_4^{\surd}$.

Then it follows that,

$M, w \models \langle T_4, 1 \rangle \alpha \to \langle T_4 \| T_5, 2 \rangle \alpha$.

Axiom system

- $\langle h, i \rangle \alpha \equiv$?

(Informally): Game h is enabled and player i has a strategy given by a sub-tree X of the enabled game such that *frontier*(X) satisfies α.

Axiom system

- $\langle h, i \rangle \alpha \equiv h^{\surd} \wedge \downarrow_{(h,i,\alpha)}$ (push)

(Informally): Game h is enabled and player i has a strategy given by a sub-tree X of the enabled game such that *frontier*(X) satisfies α.

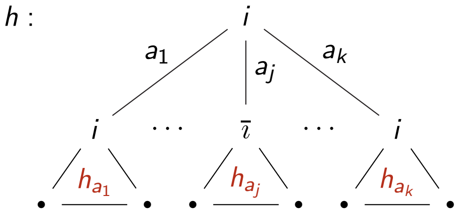
- $\langle a \rangle \alpha$ - can be encoded in the logic.
- $h^{\surd}$ can be defined.

Definition of *push*

h is a single node:

- $\downarrow_(h, i, \alpha) = \alpha$.

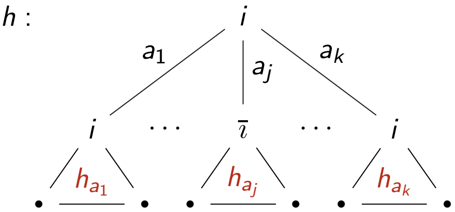
Axiom system



$\downarrow_{(}\ h, i, \alpha)$ holds at state u:

- if u is a player i node
  - then $\exists w$ such that $u \xrightarrow{a} w$ and $\langle h_a, i\rangle\alpha$ holds at w.

Axiom system



$\downarrow_{(}\ h, i, \alpha)$ holds at state u:

- if u is a player i node
  - then $\exists w$ such that $u \xrightarrow{a} w$ and $\langle h_a, i \rangle \alpha$ holds at w.

- if u is node taken by a player in $\bar{\imath}$
  - $\forall a_j$ such that $u \xrightarrow{a_j} w$, $\langle h_{a_j}, i \rangle \alpha$ holds at w.

Axiom system

General idea behind *push*

- If the root is a player i-node then
  - an edge is chosen and the requirement is "pushed" to the relevant subtree.
- If the root is a node taken by a player in $\bar{\imath}$ then
  - all outgoing edges need to be taken into account and the requirement is "pushed" to all the resulting subtrees.

Axiom system

- Propositional axioms:
  - All the substitutional instances of tautologies of PC.
  - $\mathbf{turn}_i \equiv \bigwedge_{j \in \bar{\imath}} \neg \mathbf{turn}_j$.
- Axiom for single edge games:
  - $\langle a \rangle (\alpha_1 \vee \alpha_2) \equiv \langle a \rangle \alpha_1 \vee \langle a \rangle \alpha_2$.
  - $\langle a \rangle turn_i \supset [a] turn_i$.
- Reduction axioms:
  - $\langle g_1 \cup g_2, i \rangle \alpha \equiv \langle g_1, i \rangle \alpha \vee \langle g_2, i \rangle \alpha$.
  - $\langle g_1 ; g_2, i \rangle \alpha \equiv \langle g_1, i \rangle \langle g_2, i \rangle \alpha$.
  - $\langle g_1 \| g_2, i \rangle \alpha \equiv \bigvee_{h \in init(g_1 \| g_2)} head_h^{\checkmark} \wedge comp^i(h, g_1, g_2, \alpha)$

Inference rules

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

Parallel reduction axiom

Modelling interleaving of players.

Init and Residue

- Init: $init(g)$ = the initial (atomic) game of g.
- Residue: $g \backslash h$ = the game expression generated after playing the initial atomic game h in $init(g)$.

Parallel reduction axiom

Modelling interleaving of players.

Init and Residue

- Init: $\mathit{init}(g) =$ the initial (atomic) game of g.
- Residue: $g \backslash h =$ the game expression generated after playing the initial atomic game h in $\mathit{init}(g)$.

Parallel reduction axiom: intuitive idea

- There exists an atomic tree $h \in \mathit{init}(g_1 \| g_2)$ such that $\mathit{head}(h)$ is enabled.
- Player i has a strategy in $\mathit{head}(h)$ which when composed with a strategy in the residue ensures α. We use $\mathit{comp}^i(h, g_1, g_2, \alpha)$ to denote this property.

Expressing $comp^i(h, g_1, g_2, \alpha)$

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

If $h \in init(g_1)$, $h \in init(g_2)$ and,

- $\widehat{\lambda}(s_0) = i$ then $comp^i(h, g_1, g_2, \alpha) =$
  $\bigvee_{a_j \in A}(\langle a_j\rangle\langle(h_{a_j}; (g_1 \backslash h))\| g_2\rangle\alpha \vee \langle a_j\rangle\langle g_1\|(h_{a_j}; (g_2 \backslash h))\rangle\alpha)$.
- $\widehat{\lambda}(s_0) \in \bar{\imath}$ then $comp^i(h, g_1, g_2, \alpha) =$
  $\bigwedge_{a_j \in A}([a_j]\langle(h_{a_j}; (g_1 \backslash h))\| g_2\rangle\alpha \vee [a_j]\langle g_1\|(h_{a_j}; (g_2 \backslash h))\rangle\alpha)$.

Decidability

A formula is satisfiable iff it is satisfiable in an exponential sized model.

Given α to decide if α is satisfiable:

- Guess an exponential sized model M.
- Explicitly build the relation $R_g^i \subseteq 2^{(W \times W)^*}$.
  - Time: exponential in the size of the model.
- Check whether M satisfies α.

Coalition logic

Language:

$\phi := \bot \mid p \mid \neg\phi \mid \phi \vee \phi \mid [C]\phi$

Coalition logic

Language:
$\phi := \bot \mid p \mid \neg\phi \mid \phi \vee \phi \mid [C]\phi$

Coalitional model: $\mathcal{M} = (S, \{E_C \mid C \subseteq N\}, V)$, where
$E_C : S \longrightarrow \mathcal{P}(\mathcal{P}(S))$ is monotonic.

Coalition logic

Language:
$\phi := \bot \mid p \mid \neg\phi \mid \phi \vee \phi \mid [C]\phi$

Coalitional model: $\mathcal{M} = (S, \{E_C \mid C \subseteq N\}, V)$, where
$E_C : S \longrightarrow \mathcal{P}(\mathcal{P}(S))$ is monotonic.

$\mathcal{M}, s \models [C]\phi$ iff sE_CX, where $X = \{x \in S : \mathcal{M}, x \models \varphi\}$

$[C]\phi$ holds at a state s iff coalition C is effective for the truth-set of ϕ.

Which book to read?

- A father of 3 daughters would like to read a book on the relationship between sexes.
- He wants his daughters' opinion regarding which book to read.
- Simon de Beauvoir's "Le deuxième sexe" or Susan Faludi's "Backlash".
- The majority opinion will decide.

Which book to read?

$$\varphi = [\{1,2\}]b \wedge [\{2,3\}]b$$

Which book to read?

$\varphi = [\{1, 2\}]b \wedge [\{2, 3\}]b$

$S = \{s, b, f\}$

$E(s)$

	B	F
B	b	b
F	b	f

	B	F
B	b	f
F	f	f

Daughter 1 is the row player, 2 is the column player and 3 decides between the tables.

Model checking

Given $\mathcal{M}$ and ϕ, find $\phi^{\mathcal{M}}$

$\phi^{\mathcal{M}}$ can be calculated in time $\mathcal{O}(|\mathcal{M}| \times |\phi|)$

Axiomatics

Mon

- propositional tautologies,
- $$\frac{\varphi,\ \varphi \to \psi}{\psi}$$
- $$\frac{\varphi \to \psi}{[C]\varphi \to [C]\psi}$$

Satisfiability

Given a formula ϕ, does there exist a model $\mathcal{M}$ and a state s, such that $\mathcal{M}, s \models \phi$

Result : *Mon* is NP-complete.