Logic and Games

Soumya Paul

Dept. of Theoretical Computer Science
The Institute of Mathematical Sciences
CIT Campus, Taramani
Chennai-600113

Outline

First order logic

Outline

Outline

Outline

Outline

Logic: applications

- Art of reasoning and deductions based on premises and facts
- Formalise mathematical reasoning
- Design circuits and chips
- Specify requirements for programs (hardware/software)

Logic: flavour

- Statement-1 and Statement-2
- Statement-1 or (Statement-2 and Statement-3)
- Statement-1 implies Statement-2
- For all ϵ there exists δ such that Statement

First order logic

- $\forall x \forall y \exists z (x < y \supset (x < z \wedge z < y))$

First order logic

- $\forall x \forall y \exists z (x < y \supset (x < z \wedge z < y))$
- Holds in $\mathbb{N}$?

First order logic

- $\forall x \forall y \exists z (x < y \supset (x < z \wedge z < y))$
- Holds in $\mathbb{N}$? No

First order logic

- $\forall x \forall y \exists z (x < y \supset (x < z \wedge z < y))$
- Holds in $\mathbb{N}$? No
- Holds in $\mathbb{Q}$?

First order logic

- $\forall x \forall y \exists z (x < y \supset (x < z \wedge z < y))$
- Holds in $\mathbb{N}$? No
- Holds in $\mathbb{Q}$? Yes

First order logic

- $\forall x \forall y \exists z (x < y \supset (x < z \wedge z < y))$
- Holds in $\mathbb{N}$? No
- Holds in $\mathbb{Q}$? Yes
- $(\mathbb{N}, <), (\mathbb{Q}, <), \ldots$ are structures

First order logic

- $\forall x \forall y \exists z (x < y \supset (x < z \wedge z < y))$
- Holds in $\mathbb{N}$? No
- Holds in $\mathbb{Q}$? Yes
- $(\mathbb{N}, <), (\mathbb{Q}, <), \ldots$ are structures

Model checking question: Given a structure and a first order formula, does the formula hold in the structure?

First order logic formalised

- Logical symbols: $(,), \supset, \neg, =$
- A countable set of variables: $x, y, z, \ldots$
- Quantifier symbols: $\forall, \exists$
- A countable set of predicate symbols of different arities: $p, q, r, \ldots$
- A set (possibly empty) of constant symbols

First order logic formalised - sentences

- First order sentences are defined by induction: $p, q, \neg\phi, \phi_1 \wedge \phi_2, \forall x \phi(x), \ldots$
- Sentences are evaluated on structures
- A structure is a set $\mathfrak{A}$ with an interpretation of the predicate symbols and constant symbols.

First order logic formalised - sentences

- First order sentences are defined by induction: $p, q, \neg\phi, \phi_1 \wedge \phi_2, \forall x \phi(x), \ldots$
- Sentences are evaluated on structures
- A structure is a set $\mathfrak{A}$ with an interpretation of the predicate symbols and constant symbols. Eg. $<$ is a predicate symbol which on the structure $(\mathbb{N}, <)$ has the standard interpretation

First order logic formalised - satisfaction in a structure

Given a structure $\mathfrak{A}$ and a sentence ϕ satisfaction of ϕ in $\mathfrak{A}$, denoted $\mathfrak{A} \models \phi$ is defined inductively as:

- $\mathfrak{A} \models p$ iff p holds true in $\mathfrak{A}$
- $\mathfrak{A} \models \neg\phi$ iff $\mathfrak{A} \not\models \phi$
- $\mathfrak{A} \models \phi_1 \wedge \phi_2$ iff $\mathfrak{A} \models \phi_1$ and $\mathfrak{A} \models \phi_2$
- $\mathfrak{A} \models \forall x \phi(x)$ iff for all $a \in \mathfrak{A}$, $\mathfrak{A} \models \phi[x/a]$
- $\mathfrak{A} \models \exists x \phi(x)$ iff there exists $a \in \mathfrak{A}$ such that $\mathfrak{A} \models \phi[x/a]$

First order logic formalised - satisfaction in a structure

Given a structure $\mathfrak{A}$ and a sentence ϕ satisfaction of ϕ in $\mathfrak{A}$, denoted $\mathfrak{A} \models \phi$ is defined inductively as:

- $\mathfrak{A} \models p$ iff p holds true in $\mathfrak{A}$
- $\mathfrak{A} \models \neg\phi$ iff $\mathfrak{A} \not\models \phi$
- $\mathfrak{A} \models \phi_1 \wedge \phi_2$ iff $\mathfrak{A} \models \phi_1$ and $\mathfrak{A} \models \phi_2$
- $\mathfrak{A} \models \forall x \phi(x)$ iff for all $a \in \mathfrak{A}$, $\mathfrak{A} \models \phi[x/a]$
- $\mathfrak{A} \models \exists x \phi(x)$ iff there exists $a \in \mathfrak{A}$ such that $\mathfrak{A} \models \phi[x/a]$

Model checking question: Given a structure $\mathfrak{A}$ and a first order sentence ϕ, does $\mathfrak{A} \models \phi$?

Two-player games on graphs: arenas

- Arena $\mathcal{A} = (V, E)$ such that $V = V_0 \cup V_1$ where
  - V_0 are Player 0 vertices, denoted by $\bigcirc$ and
  - V_1 are Player 1 vertices, denoted by $\square$
  - $E \subseteq V \times V$ is the edge relation

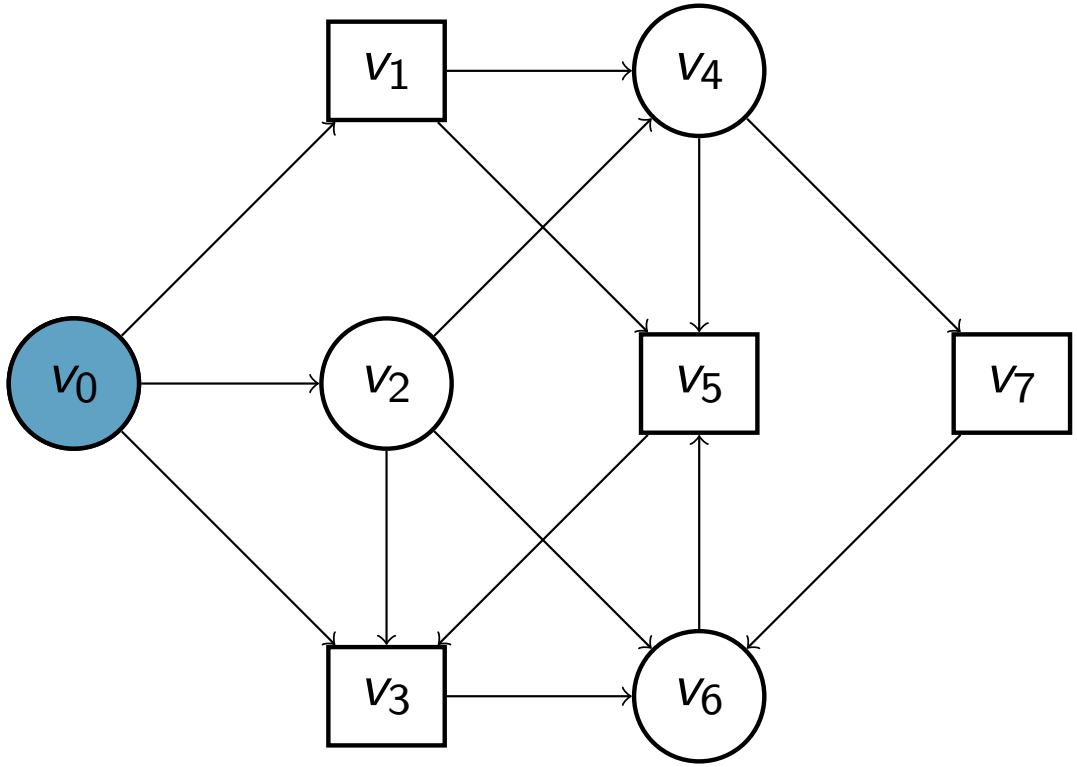
Two-player games on graphs: arenas

- Arena $\mathcal{A} = (V, E)$ such that $V = V_0 \cup V_1$ where
  - V_0 are Player 0 vertices, denoted by $\bigcirc$ and
  - V_1 are Player 1 vertices, denoted by $\square$
  - $E \subseteq V \times V$ is the edge relation
- $vE = \{u \mid (v, u) \in E\}$ are the neighbours of v

Two-player games on graphs: arenas

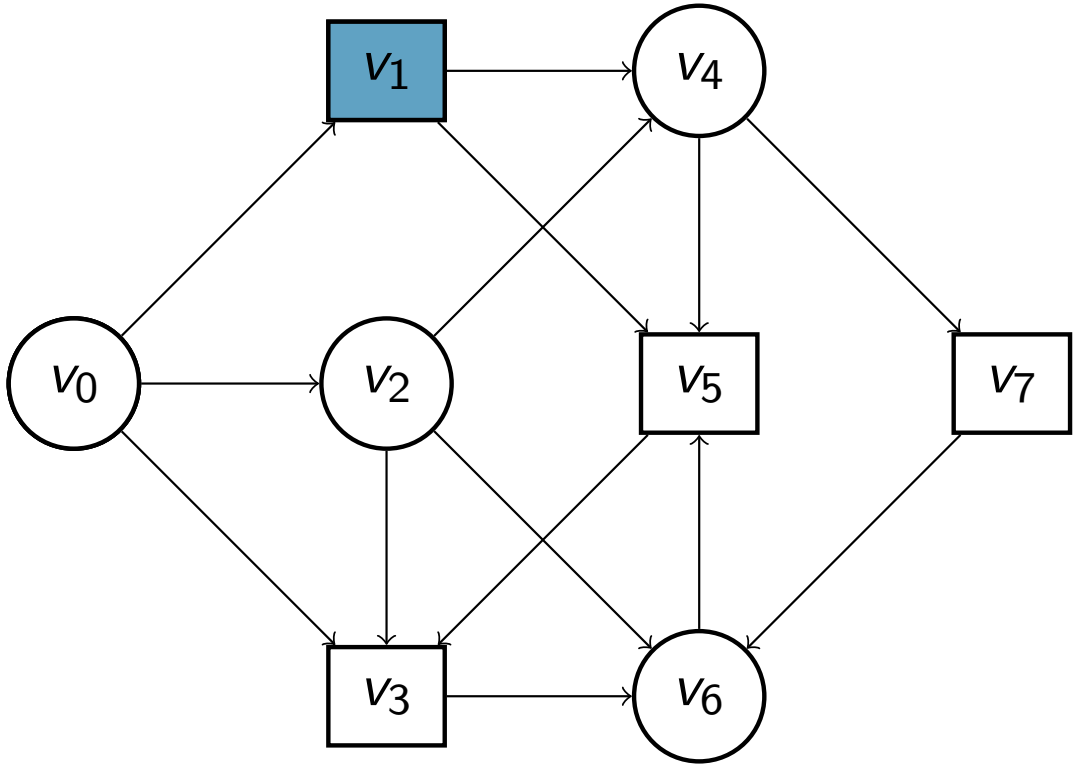
- Arena $\mathcal{A} = (V, E)$ such that $V = V_0 \cup V_1$ where
  - V_0 are Player 0 vertices, denoted by $\bigcirc$ and
  - V_1 are Player 1 vertices, denoted by $\Box$
  - $E \subseteq V \times V$ is the edge relation
- $vE = \{u \mid (v, u) \in E\}$ are the neighbours of v
- $(\mathcal{A}, v_0)$ is an initialised arena where $v_0 \in V$ is a designated vertex

Two-player games on graphs: plays



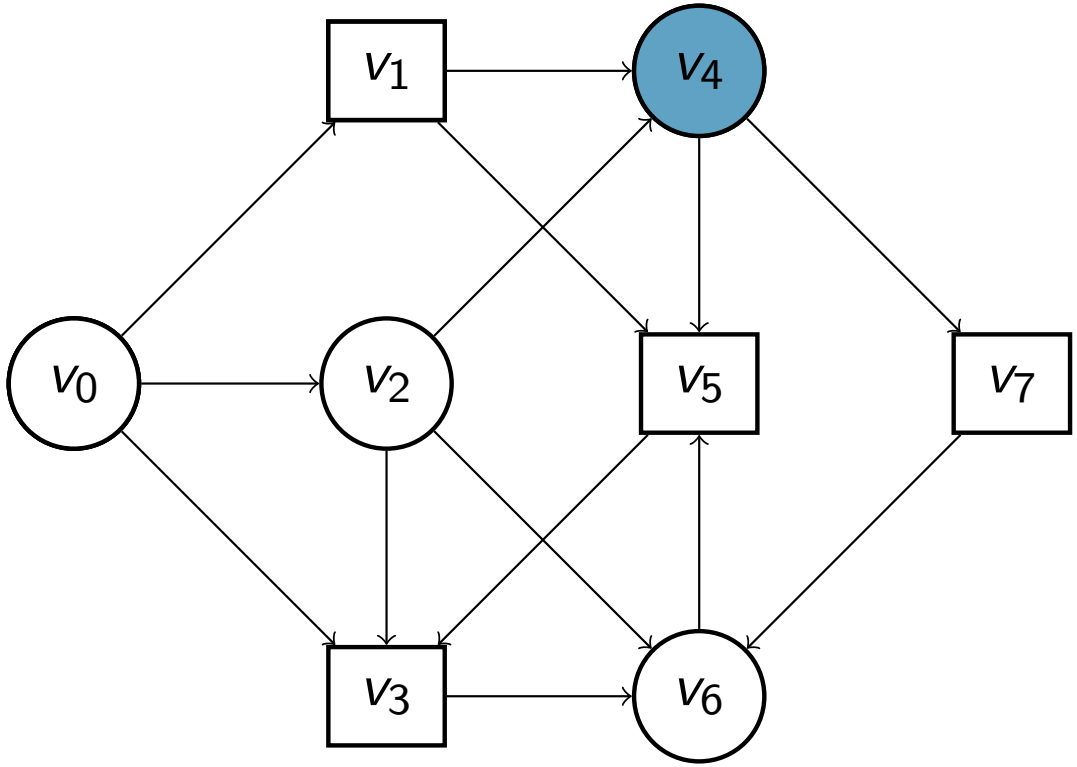
$\pi = v_0$

Two-player games on graphs: plays



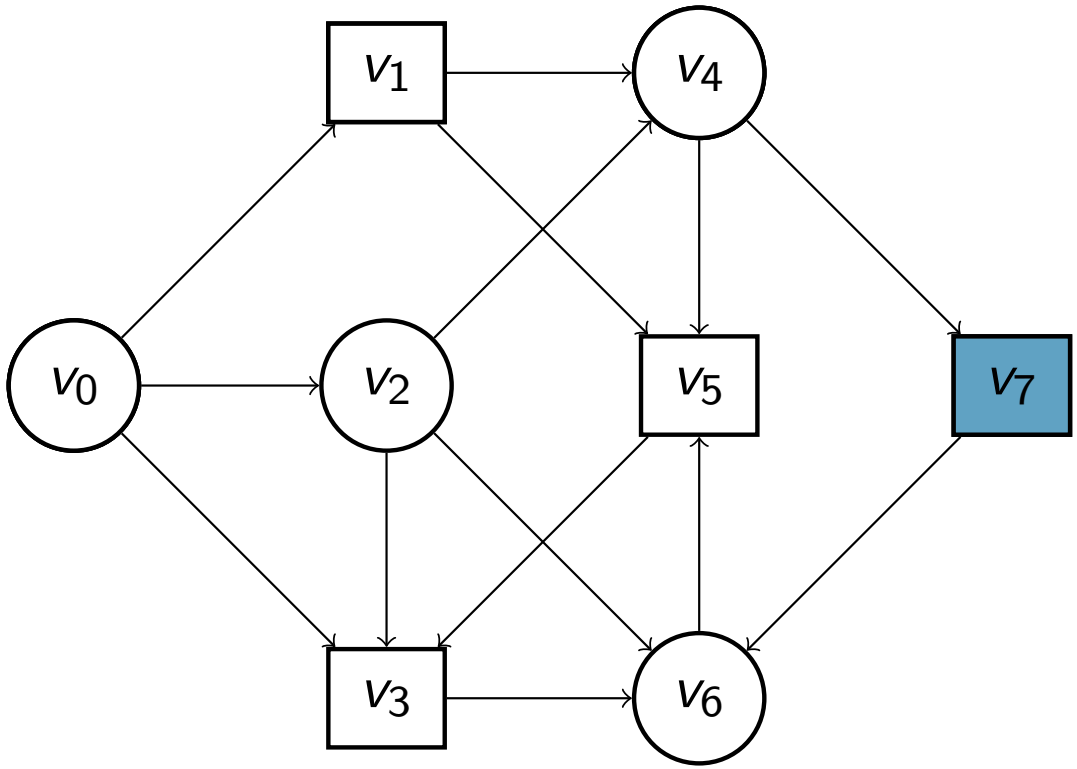
$\pi = v_0 \; v_1$

Two-player games on graphs: plays



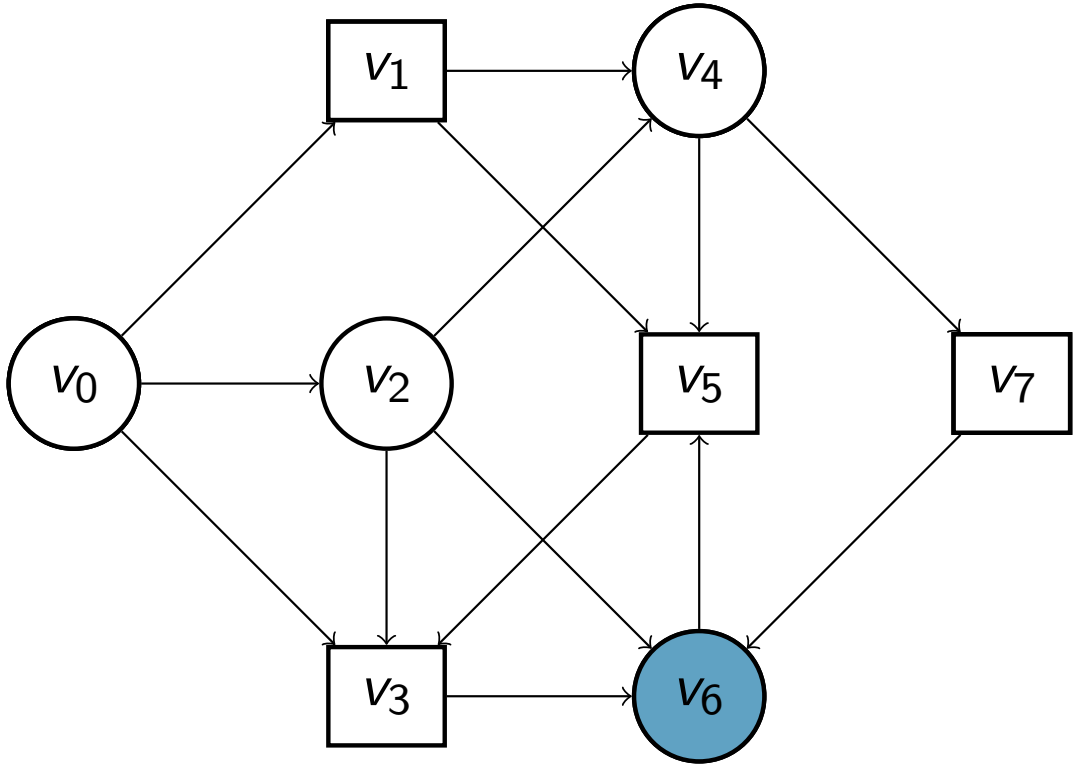
$\pi = v_0\ v_1\ v_4$

Two-player games on graphs: plays



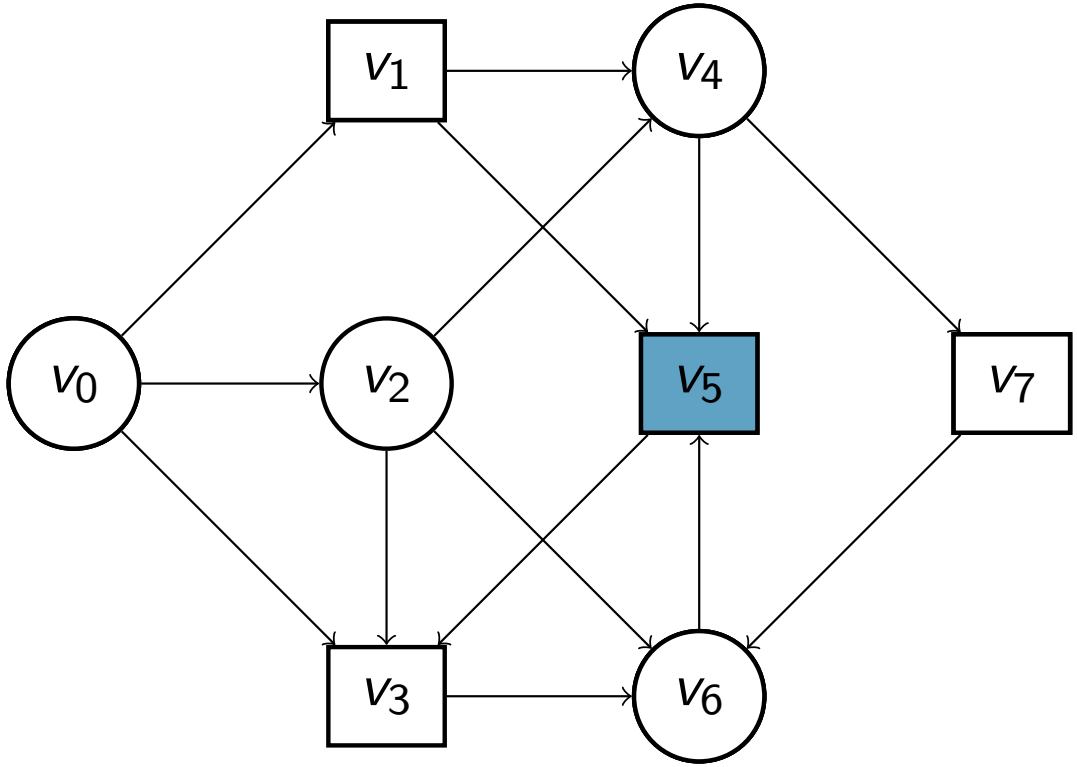
$\pi = v_0\ v_1\ v_4\ v_7$

Two-player games on graphs: plays



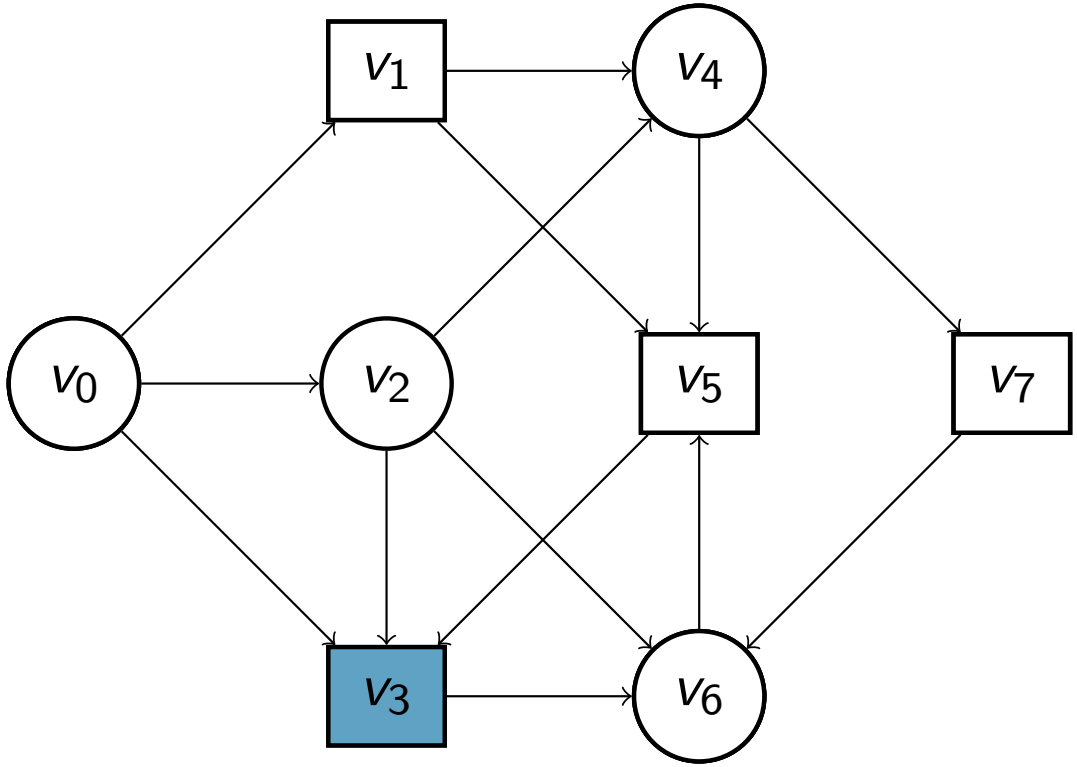
$\pi = v_0\ v_1\ v_4\ v_7\ v_6$

Two-player games on graphs: plays



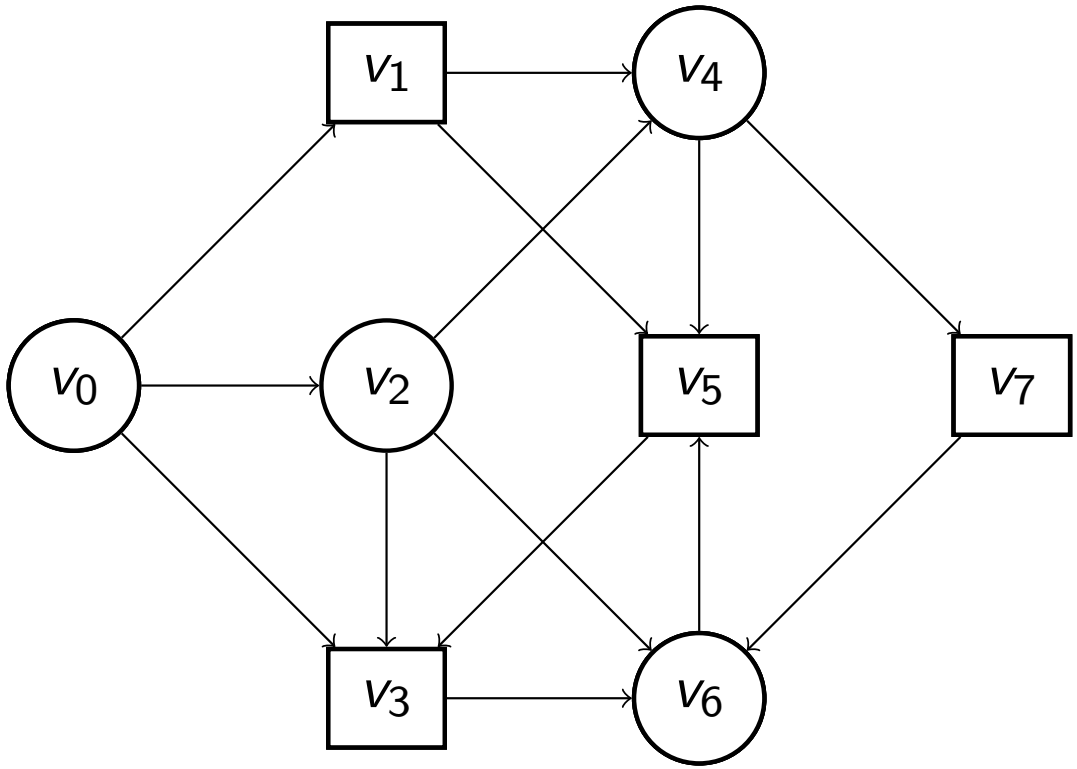
$\pi = v_0\ v_1\ v_4\ v_7\ v_6\ v_5$

Two-player games on graphs: plays



$\pi = v_0\ v_1\ v_4\ v_7\ v_6\ v_5\ v_3$

Two-player games on graphs: plays



$$\pi = v_0\ v_1\ v_4\ v_7\ v_6\ v_5\ v_3 \in V^*$$

Two-player games on graphs: strategies

- A history ρ is any finite prefix of a play π
- A strategy σ for player p, $p \in \{0, 1\}$ is a function $\sigma : V^* V_p \to V$ from the set of histories to vertices such that $\sigma(\rho v) \in vE$ for all ρ

Two-player games on graphs: strategies

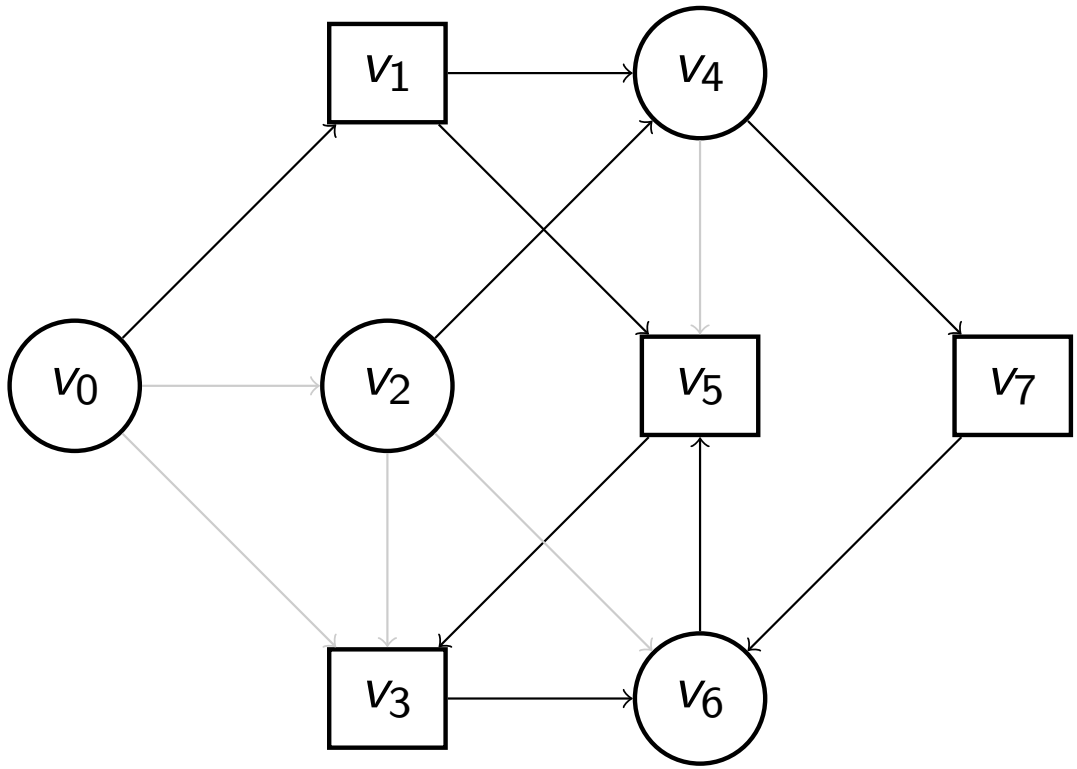
- A history ρ is any finite prefix of a play π
- A strategy σ for player p, $p \in \{0, 1\}$ is a function $\sigma : V^* V_p \to V$ from the set of histories to vertices such that $\sigma(\rho v) \in vE$ for all ρ
- A winning strategy for player p is a strategy such that by playing according to it, she can always win, no matter how the other player plays

Two-player games on graphs: strategies

- A history ρ is any finite prefix of a play π
- A strategy σ for player p, $p \in \{0, 1\}$ is a function $\sigma : V^* V_p \to V$ from the set of histories to vertices such that $\sigma(\rho v) \in vE$ for all ρ
- A winning strategy for player p is a strategy such that by playing according to it, she can always win, no matter how the other player plays
- A strategy σ for player p is memoryless or positional if it does not depend on the history. That is $\sigma : V_p \to V$

Memoryless strategy

Memoryless strategy as a subgraph



Model checking game for first order logic: arena

Given a structure $\mathfrak{A}$ and a first order sentence ϕ we construct an arena $G(\mathfrak{A}, \phi)$ as follows:

- Played between two players: Alter($\forall$) and Ego($\exists$)
- The vertices of $G(\mathfrak{A}, \phi)$ are subformulas of ϕ and subformulas of ϕ with variables substituted with elements from $\mathfrak{A}$
- The initial vertex is labelled with the formula ϕ
- The players at each vertex and the moves of the players are defined depending on the topmost connective of the current subformula

Model checking game for first order logic: moves

- $\phi_1 \vee \phi_2$: Has two neighbours ϕ_1 and ϕ_2. Player Alter chooses one of the two neighbours
- $\phi_1 \wedge \phi_2$: Has two neighbours ϕ_1 and ϕ_2. Player Ego chooses one of the two neighbours
- $\neg\phi$: Has one neighbour ϕ. Player Ego chooses ϕ and the roles of the players are interchanged
- $\forall x \phi(x)$: Has $|\mathfrak{A}|$ neighbours, one neighbour $\phi[x/a]$ for each $a \in \mathfrak{A}$. Player Alter chooses $a \in \mathfrak{A}$ and the play moves along the corresponding edge to vertex $\phi[x/a]$
- $\exists x \phi(x)$: Has $|\mathfrak{A}|$ neighbours, one neighbour $\phi[x/a]$ for each $a \in \mathfrak{A}$. Player Ego chooses $a \in \mathfrak{A}$ and the play moves along the corresponding edge to vertex $\phi[x/a]$
- The terminal vertices consist of the predicates

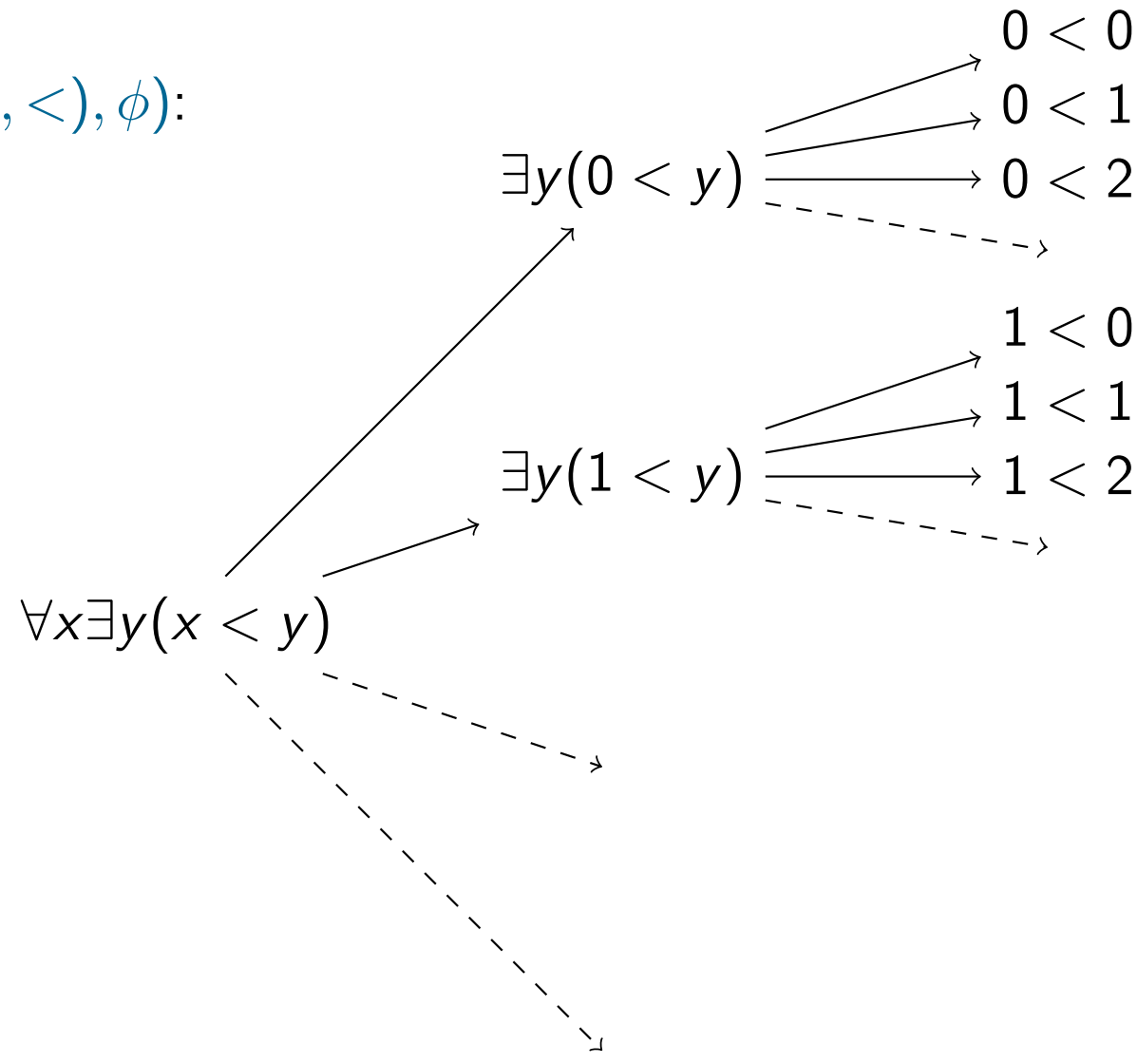
Model checking game for first order logic: winning condition

- Player Ego wins the game if at a terminal vertex p, p holds true of $\mathfrak{A}$
- Player Alter wins otherwise

Model checking game for first order logic: example

$\phi \equiv \forall x \exists y (x < y)$ on $(\mathbb{N}, <)$

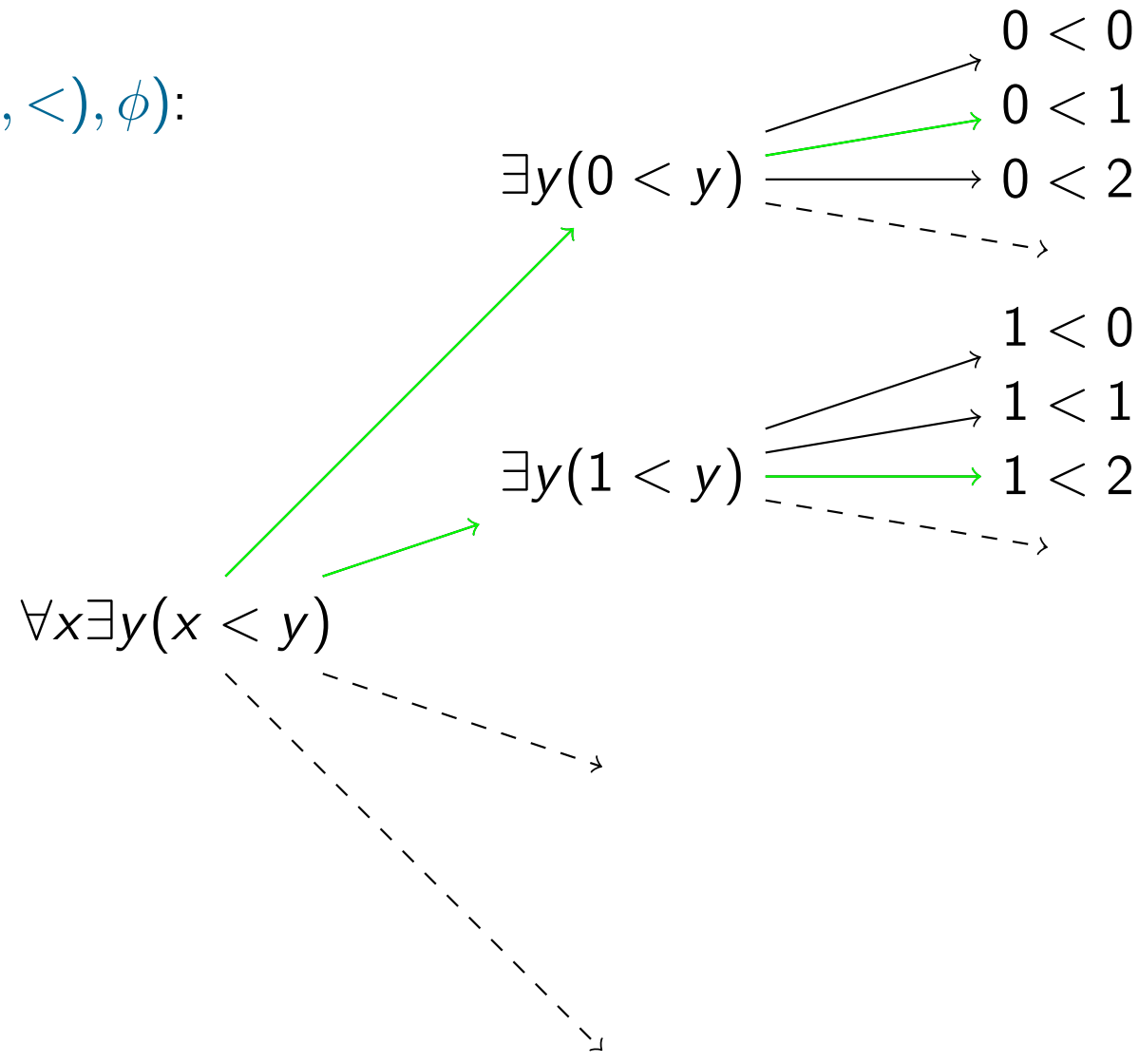
$G((\mathbb{N}, <), \phi)$:



Model checking game for first order logic: example

$\phi \equiv \forall x \exists y (x < y)$ on $(\mathbb{N}, <)$

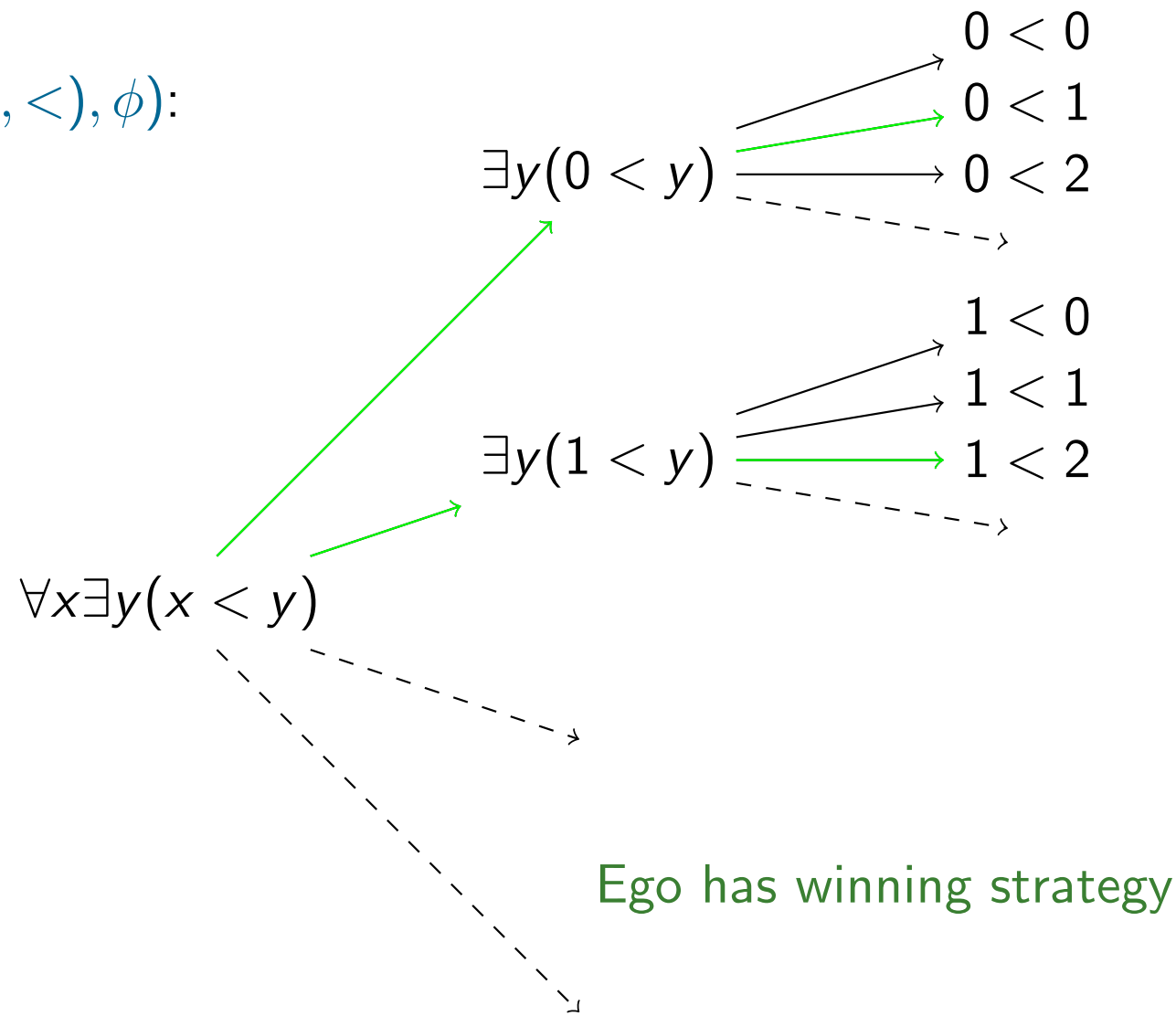
$G((\mathbb{N}, <), \phi)$:



Model checking game for first order logic: example

$\phi \equiv \forall x \exists y (x < y)$ on $(\mathbb{N}, <)$

$G((\mathbb{N}, <), \phi)$:

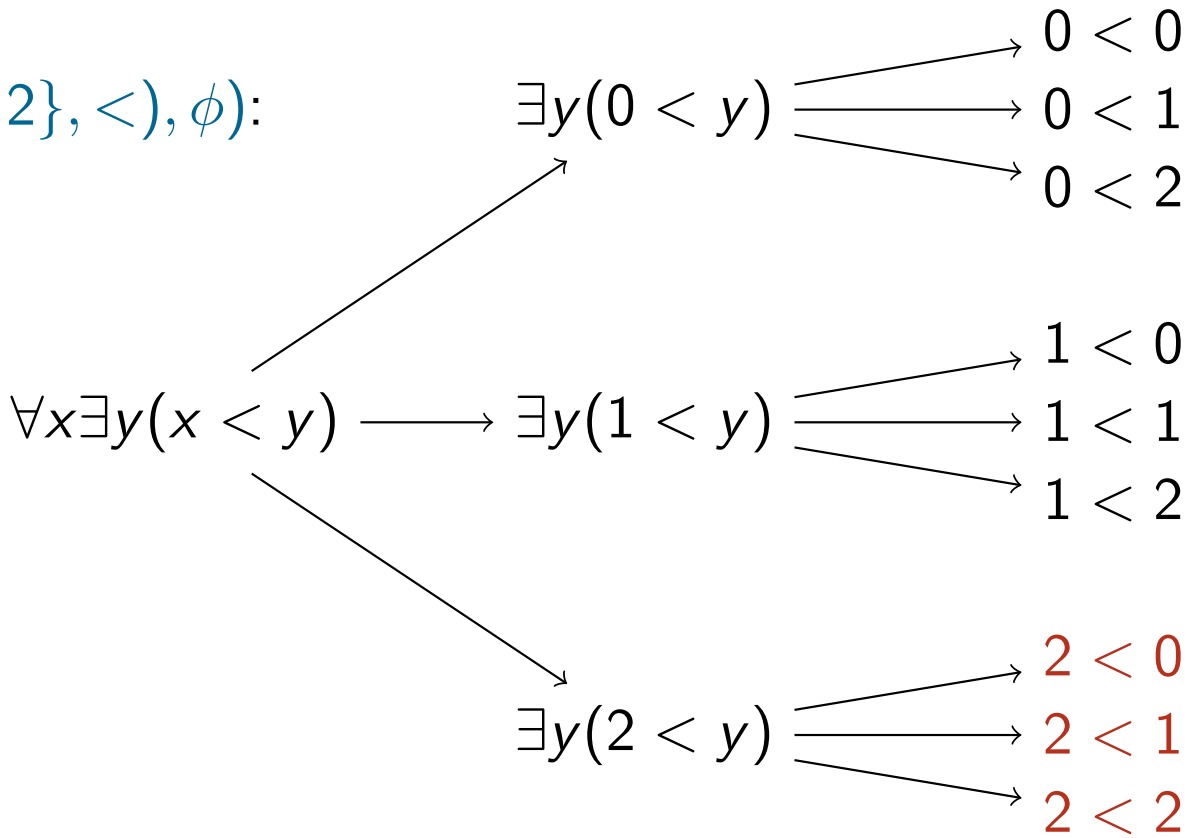


Ego has winning strategy

Model checking game for first order logic: example

$\phi \equiv \forall x \exists y (x < y)$ on $(\{0, 1, 2\}, <)$

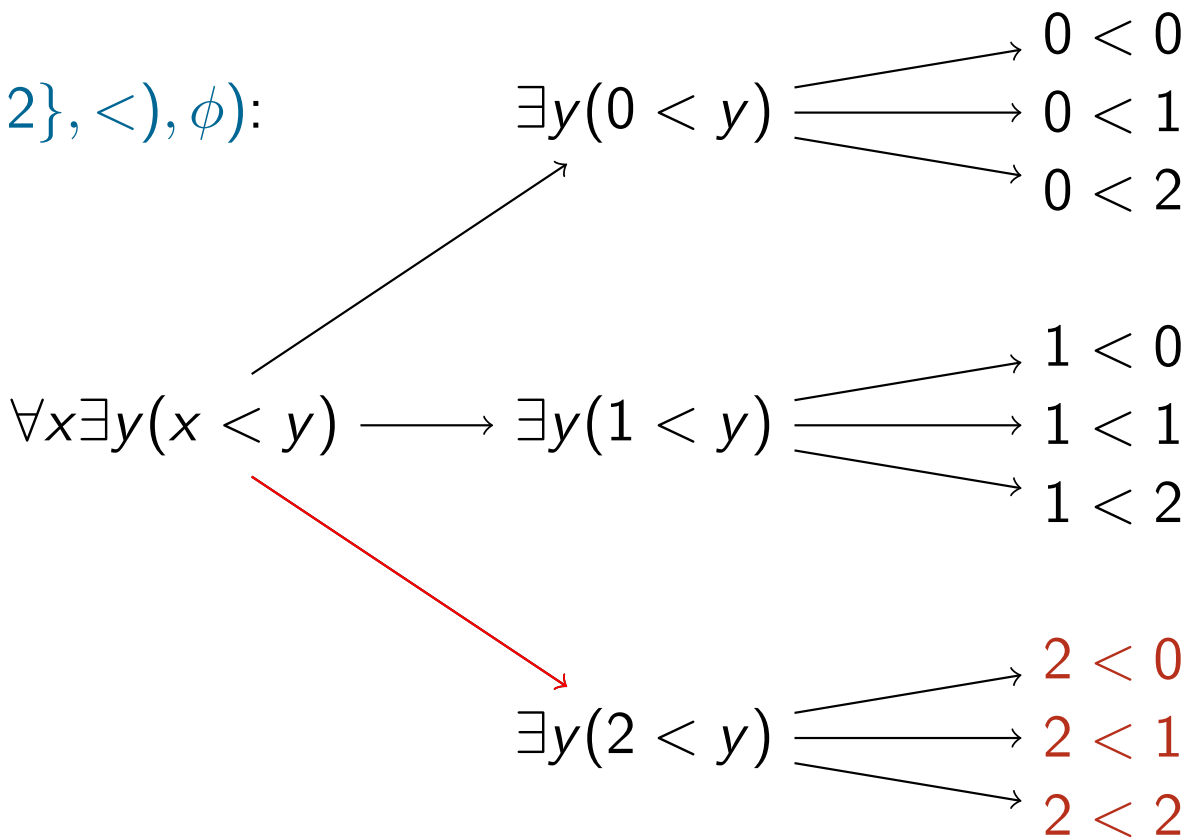
$G((\{0, 1, 2\}, <), \phi)$:



Model checking game for first order logic: example

$\phi \equiv \forall x \exists y (x < y)$ on $(\{0, 1, 2\}, <)$

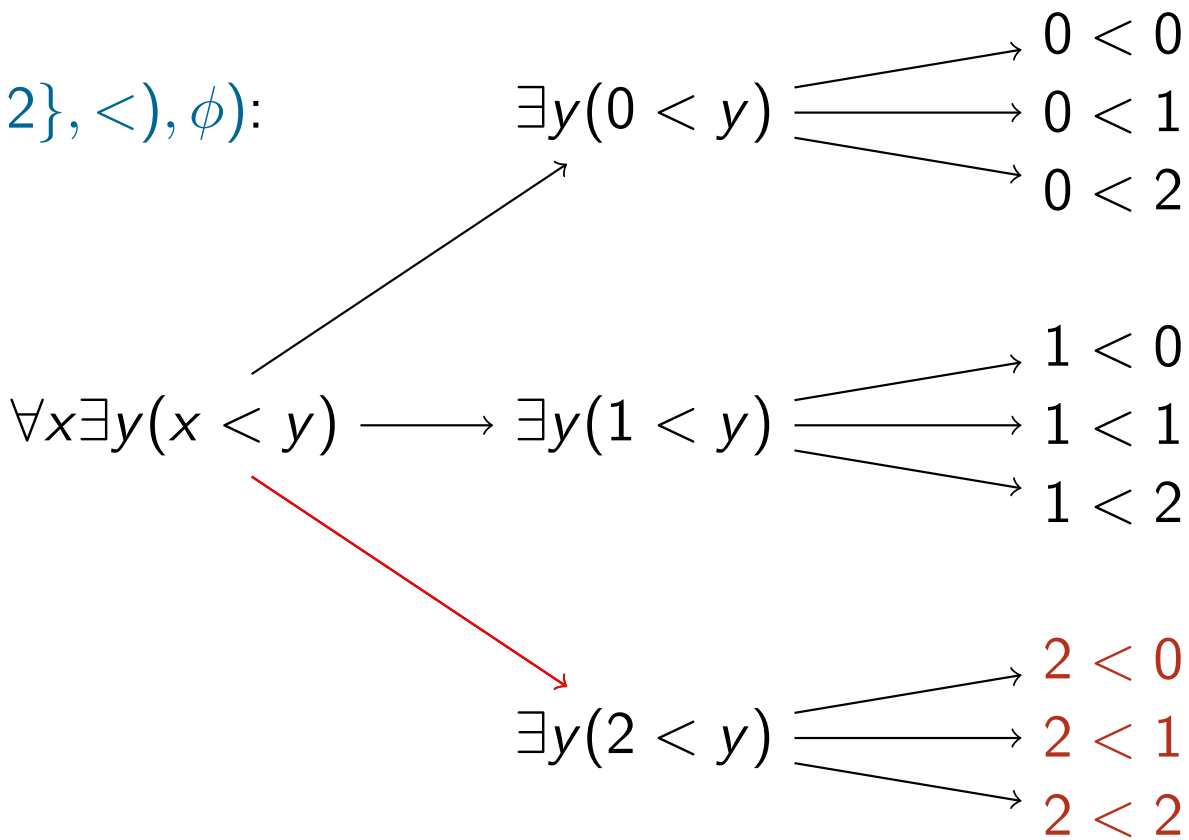
$G((\{0, 1, 2\}, <), \phi)$:



Model checking game for first order logic: example

$\phi \equiv \forall x \exists y (x < y)$ on $(\{0, 1, 2\}, <)$

$G((\{0, 1, 2\}, <), \phi)$:



Alter has a winning strategy

The strategy theorem

Theorem

Given a structure $\mathfrak{A}$ and a first order sentence ϕ, $\mathfrak{A} \models \phi$ if and only if Ego has a winning strategy in the game $G(\mathfrak{A}, \phi)$

The strategy theorem

Theorem

Given a structure $\mathfrak{A}$ and a first order sentence ϕ, $\mathfrak{A} \models \phi$ if and only if Ego has a winning strategy in the game $G(\mathfrak{A}, \phi)$

Proof.

Induct on the structure of ϕ

- p: Base case. $\mathfrak{A} \models p$ implies Ego wins the single vertex game $G(\mathfrak{A}, p)$
- $\phi_1 \vee \phi_2$: $\mathfrak{A} \models \phi_1 \vee \phi_2$ iff $\mathfrak{A} \models \phi_1$ or $\mathfrak{A} \models \phi_2$ iff Ego has a winning strategy from ϕ_1 or ϕ_2
- $\phi_1 \wedge \phi_2$: $\mathfrak{A} \models \phi_1 \wedge \phi_2$ iff $\mathfrak{A} \models \phi_1$ and $\mathfrak{A} \models \phi_2$ iff Ego has a winning strategy from ϕ_1 and ϕ_2

The strategy theorem: proof (contd.)

Proof.

- $\neg\phi$: $\mathfrak{A} \models \neg\phi$ iff $\mathfrak{A} \not\models \phi$ iff Ego does not have a winning strategy from ϕ iff Alter has a winning strategy from ϕ
- $\forall x\phi(x)$: $\mathfrak{A} \models \forall x\phi(x)$ iff for all $a \in \mathfrak{A}$, $\mathfrak{A} \models \phi[x/a]$ iff Ego has a winning strategy from $\phi[x/a]$
- $\exists x\phi(x)$: $\mathfrak{A} \models \exists x\phi(x)$ iff there exists $a \in \mathfrak{A}$ such that $\mathfrak{A} \models \phi[x/a]$ iff Ego has a winning strategy from $\phi[x/a]$

Other logics

- First order logic - perfect recall. Independence friendly logic (Hintikka, Sandu) - imperfect recall. Only semantics via games
- Modal logic - evaluated on Kripke frames. One can prove similar strategy theorem
- μ-calculus and its fragments - games on graphs with infinite plays. Parity winning condition

Importance of game semantics

- Helps gain insight into the logic. Logic is 'well-defined' if it admits a natural game semantics
- Extremely useful in verification and model checking of programs (hardware/software).

Importance of game semantics

- Helps gain insight into the logic. Logic is 'well-defined' if it admits a natural game semantics
- Extremely useful in verification and model checking of programs (hardware/software).Verifying the correctness of a piece of hardware or software amounts to checking whether one of the players has a winning strategy in a certain game and its construction

Importance of game semantics

- Helps gain insight into the logic. Logic is 'well-defined' if it admits a natural game semantics
- Extremely useful in verification and model checking of programs (hardware/software).Verifying the correctness of a piece of hardware or software amounts to checking whether one of the players has a winning strategy in a certain game and its construction

Thanks for your attention!