First-Order Logic with Counting for General Game Playing

Łukasz Kaiser and Łukasz Stafiniak

CNRS & LIAFA, Paris

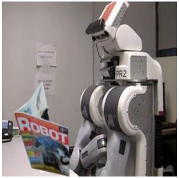
AAAI 2011, San Francisco

Motivation

Goal: demonstrate any board game to ~~a robot~~ and have him play it well

(1) Demonstration

→



(2) Description

→

(3) Learning

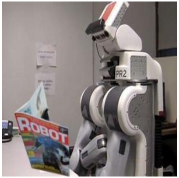
→

(4) Play

Motivation

Goal: demonstrate any board game and play it well

→

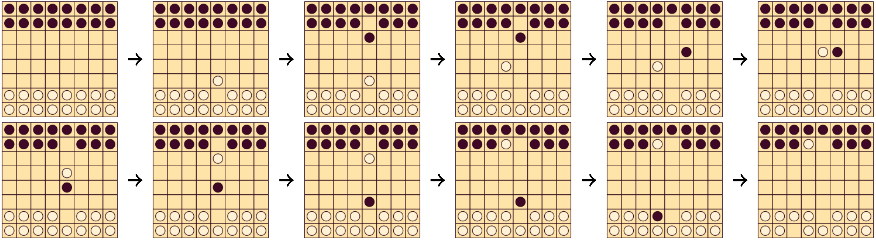


→

→

(1) Demonstration (2) Description (3) Learning (4) Play

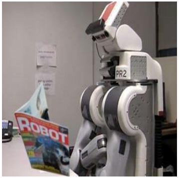
Example Input: Breakthrough moves, a **sequence of screenshots**



Motivation

Goal: demonstrate any board game and play it well

(1) Demonstration

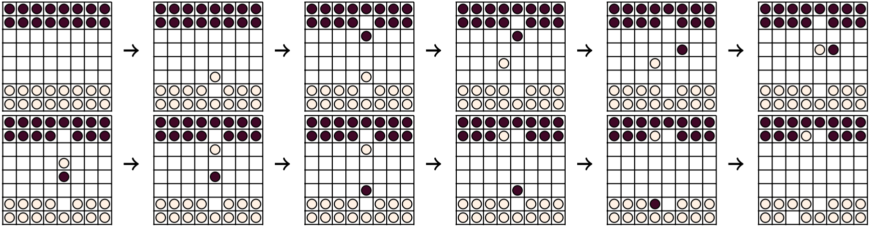


(2) Description

(3) Learning

(4) Play

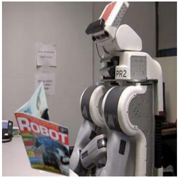
Derived Structures: segmentation of the input, row and column relations



Motivation

Goal: demonstrate any board game and play it well

→

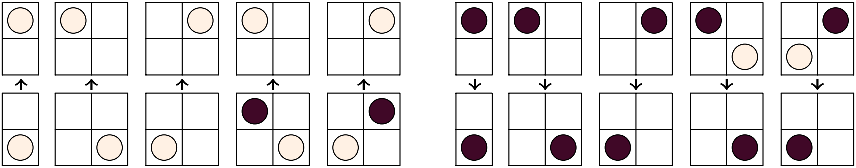


→

→

(1) Demonstration (2) Description (3) Learning (4) Play

Derived Structure Rewriting Rules correspond **directly** to moves

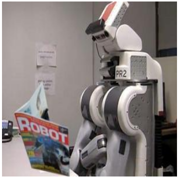


Motivation

Goal: demonstrate any board game and play it well

(1) Demonstration

→



(2) Description

→

(3) Learning

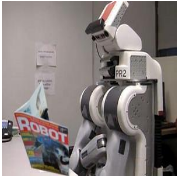
→

(4) Play

Why that? You cannot demonstrate **everything** – sometimes you **say it**.

Motivation

Goal: demonstrate any board game and play it well

 → 

→  → 

(1) Demonstration | **(2) Description** | (3) Learning | (4) Play

Why that? You cannot demonstrate **everything** – sometimes you **say it**.

Example: Breakthrough Winning Condition for White

Text: **Some white piece must be in the last row.**

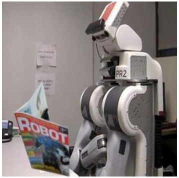
Last row is the one for which there is no next row.

Formula: $\exists x\ \mathbf{White}(x) \wedge \mathbf{LastRow}(x)$

$\mathbf{LastRow}(x) \equiv \neg \exists y\ \mathbf{NextRow}(x, y)$

Motivation

Goal: demonstrate any board game and play it well

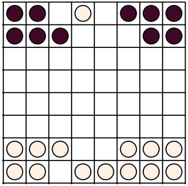

→


→

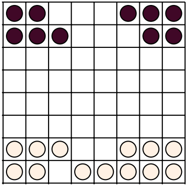
→


(1) Demonstration **(2) Description** (3) Learning (4) Play

Automatic Derivation of Simple Constraints



Positive Example



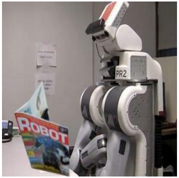
Negative Example

Derived Formula: $\exists\, x\ (\mathbf{White}(x)\ \wedge\ \forall\, y\ \neg\mathbf{NextRow}(x, y))$

Motivation

Goal: demonstrate any board game and play it well

$\longrightarrow$



$\longrightarrow$

$\longrightarrow$

(1) Demonstration (2) Description **(3) Learning** **(4) Play**

Deriving Interesting Patterns and **Learning Weights**

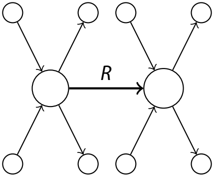
- **(i)** When I move, what do I **add or delete**? **White added, Black deleted**
- **(ii)** Can I **expand** the goal to an **existential conjunction**?

$$\exists x_1 \ldots x_8 \left(\mathbf{NextRow}(x_1, x_2) \wedge \ldots \wedge \mathbf{NextRow}(x_7, x_8) \wedge \mathbf{White}(x_8) \right)$$

- **(iii)** How many of these **conjunction items** are **realized**?
- **(iv)** Other expansions and sums over co-occurring items.
- **(v)** Use weighted sums of such patterns.

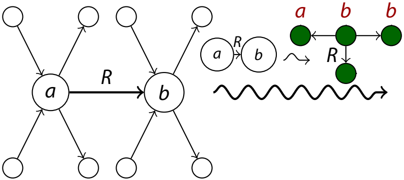
Structure Rewriting Rules

Rewriting Example



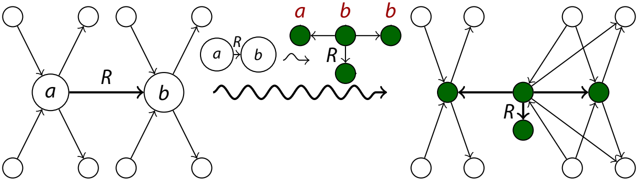
Structure Rewriting Rules

Rewriting Example



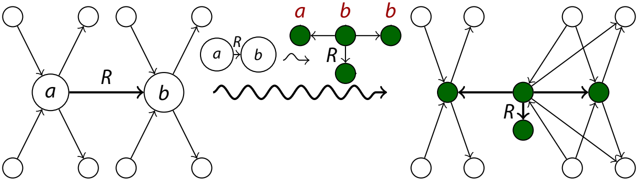
Structure Rewriting Rules

Rewriting Example



Structure Rewriting Rules

Rewriting Example



Embedding: σ is **injective** and $R_i^{\mathfrak{A}}(a_1, \dots, a_{r_i}) \Leftrightarrow R_i^{\mathfrak{B}}(\sigma(a_1), \dots, \sigma(a_{r_i}))$

$$\sigma \;:\quad \mathfrak{A} = (A, R_1^{\mathfrak{A}}, R_2^{\mathfrak{A}}, \dots, R_k^{\mathfrak{A}}) \quad \hookrightarrow \quad (B, R_1^{\mathfrak{B}}, R_2^{\mathfrak{B}}, \dots, R_k^{\mathfrak{B}}) = \mathfrak{B}$$

Rewriting: $\mathfrak{B} = \mathfrak{A}[\mathfrak{L} \to \mathfrak{R}/\sigma]$ iff $B = (A \setminus \sigma(L)) \dot\cup R$ and,
for $M = \{(r, a) \mid a = \sigma(l), r \in P_l^{\mathfrak{R}}$ for some $l \in L\} \cup \{(a, a) \mid a \in A\}$,
$(b_1, \dots, b_{r_i}) \in R_i^{\mathfrak{B}} \;\Leftrightarrow\; (b_1, \dots, b_{r_i}) \in R_i^{\mathfrak{R}}$ **or** $(b_1 M \times \dots \times b_{r_i} M) \cap R_i^{\mathfrak{A}} \neq \emptyset$.
(in the second case at least one $b_j \notin \mathfrak{A}$)

First-Order Logic with Counting

Syntax

$$
\begin{aligned}
\varphi &:= R_i(x_1, \ldots, x_{r_i}) \mid x_i = x_j \mid \rho < \rho \\
&\quad\ \mid \neg\varphi \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \exists x_i\, \varphi \mid \forall x_i\, \varphi \\
\rho &:= \frac{n}{m} \mid \rho + \rho \mid \rho \cdot \rho \mid \chi[\varphi] \mid \sum_{\overline{x} \mid \varphi} \rho
\end{aligned}
$$

Semantics as expected, with

- $\chi[\varphi(\overline{x})] = 1$ iff $\varphi(\overline{x})$ **holds**
- $\sum_{\overline{x}|\varphi} \rho$ sums $\rho(\overline{x})$ over $\overline{x}$ **satisfying** $\varphi(\overline{x})$

First-Order Logic with Counting

Syntax

$$\begin{aligned}\varphi \ &:= \ R_i(x_1, \dots, x_{r_i}) \mid x_i = x_j \mid \rho < \rho \\ &\quad\ \mid \neg\varphi \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \exists x_i\, \varphi \mid \forall x_i\, \varphi \\ \rho \ &:= \ \frac{n}{m} \mid \rho + \rho \mid \rho \cdot \rho \mid \chi[\varphi] \mid \sum_{\overline{x} \mid \varphi} \rho\end{aligned}$$

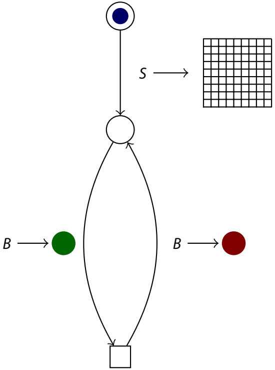
Semantics as expected, with

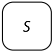
- $\chi[\varphi(\overline{x})] = 1$ iff $\varphi(\overline{x})$ **holds**
- $\sum_{\overline{x}|\varphi} \rho$ sums $\rho(\overline{x})$ over $\overline{x}$ **satisfying** $\varphi(\overline{x})$

Example from Chess

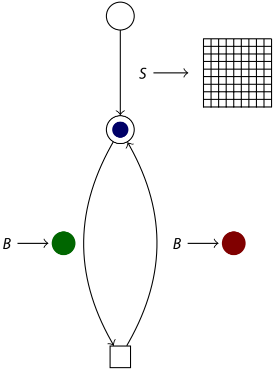
$$\sum_{x \,\mid\, \mathbf{bBeats}(x)} 1 + \chi[\mathbf{w}(x)] + 3 \cdot \chi[\mathbf{wK}(x)]$$

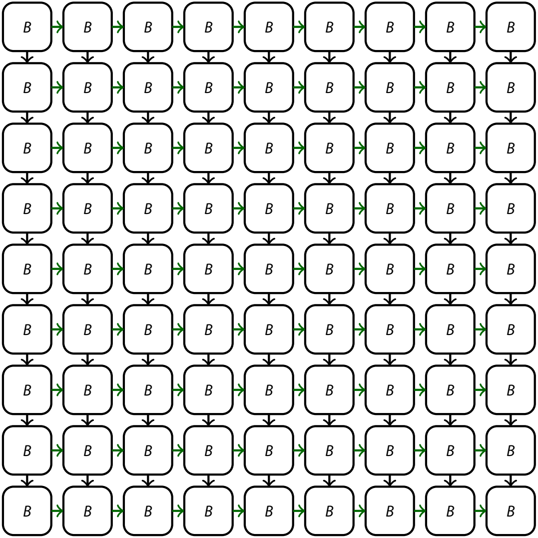
Example Game: Gomoku (Connect–5)



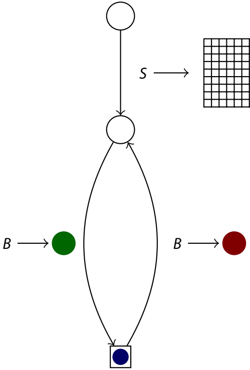


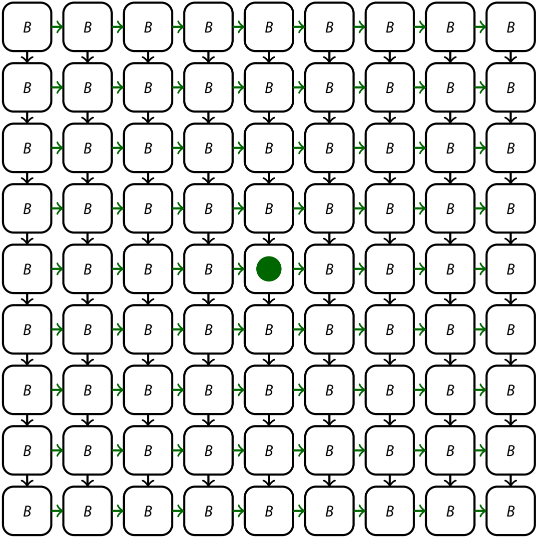
Example Game: Gomoku (Connect–5)



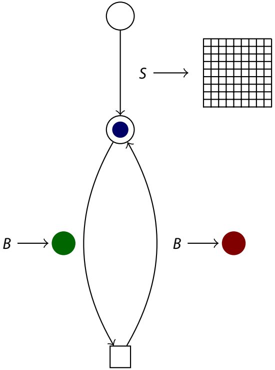


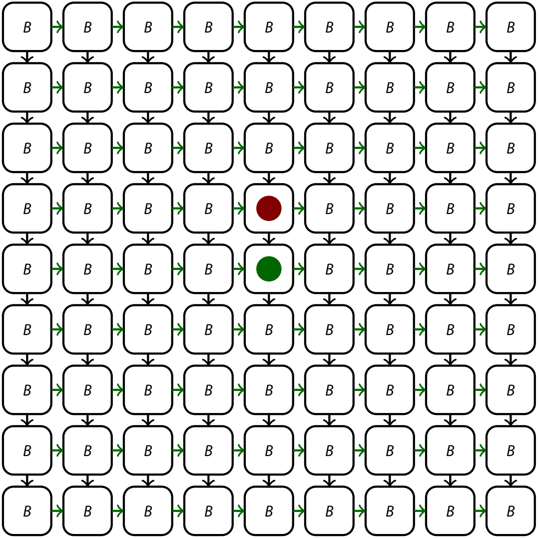
Example Game: Gomoku (Connect–5)



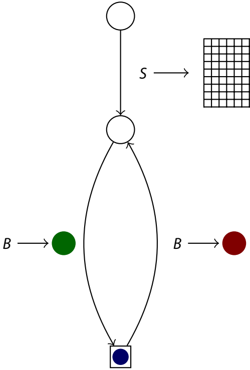


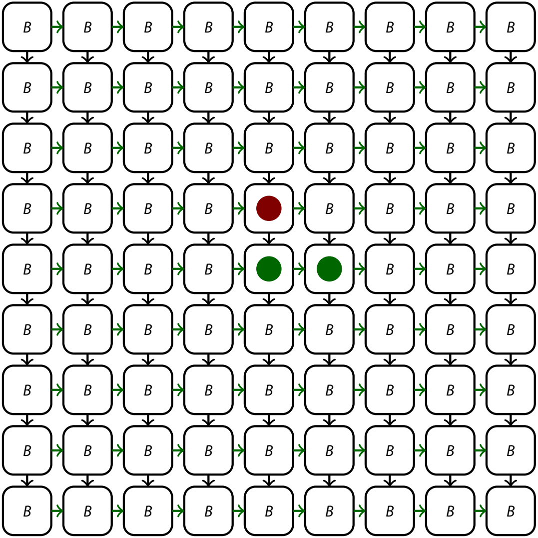
Example Game: Gomoku (Connect–5)



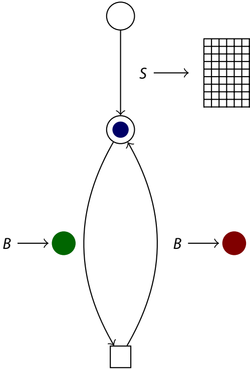


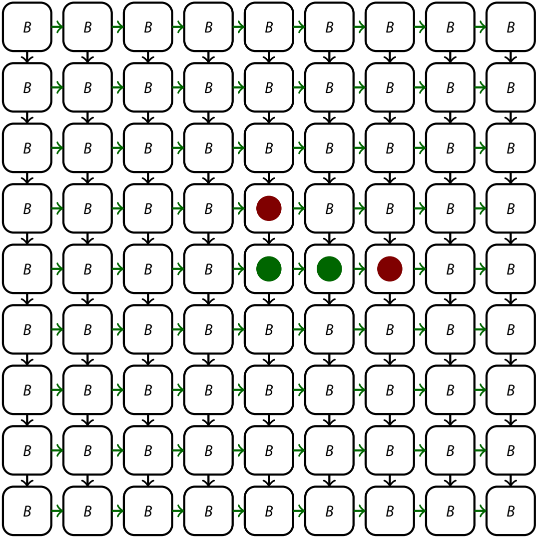
Example Game: Gomoku (Connect–5)



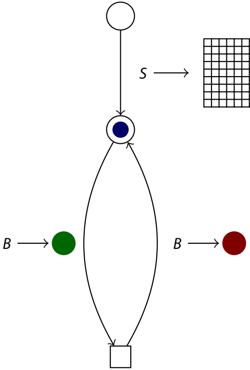


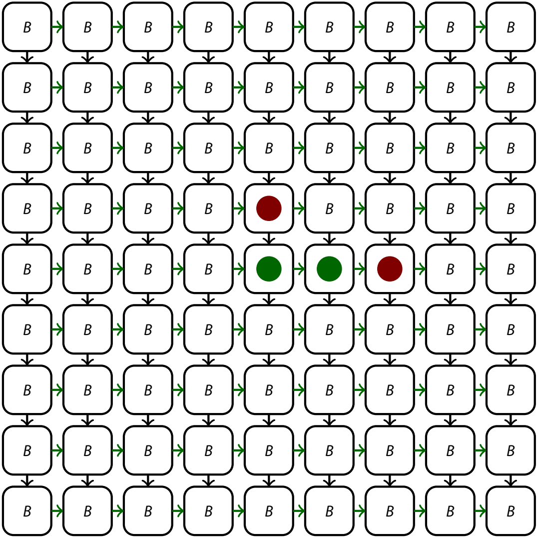
Example Game: Gomoku (Connect–5)





Example Game: Gomoku (Connect–5)





$$\chi\Big[\exists x_1 \dots x_5 \Big(\bigwedge_{1\le i\le 5} G(x_i) \wedge \Big(\bigwedge_{1\le i\le 5} R(x_i, x_{i+1}) \vee \bigwedge_{1\le i\le 5} C(x_i, x_{i+1}) \vee$$

$$\bigwedge_{1\le i\le 5} \exists y(R(x_i, y) \wedge C(y, x_{i+1})) \vee \bigwedge_{1\le i\le 5} \exists y(R(x_i, y) \wedge C(x_{i+1}, y))\Big)\Big)\Big]$$

Deriving Heuristics

Methods

- Goal expansion and Type Normal Form
- Summing over conjuncts in existentially quantified conjunctions
- Retain stable guards and include rule preconditions

Example: Tic-Tac-Toe without Diagonals

$$\big(P(x)\wedge P(y)\wedge P(z)\wedge R(x,y)\wedge R(y,z)\big)\vee\big(P(x)\wedge P(y)\wedge P(z)\wedge C(x,y)\wedge C(y,z)\big)$$

$$\rightsquigarrow$$

$$\textstyle\sum_{x|P(x)}\Big(\frac{1}{8}+\sum_{y|P(y)\wedge R(x,y)}\big(\frac{1}{4}+\sum_{z|P(z)\wedge R(y,z)}1\big)\Big)+\sum_{x|P(x)}\Big(\frac{1}{8}+\sum_{y|P(y)\wedge C(x,y)}\big(\frac{1}{4}+\sum_{z|P(z)\wedge C(y,z)}1\big)\Big)$$

Deriving Heuristics

Methods

- Goal expansion and Type Normal Form
- Summing over conjuncts in existentially quantified conjunctions
- Retain stable guards and include rule preconditions

Example: Tic-Tac-Toe without Diagonals

$$\big(P(x)\wedge P(y)\wedge P(z)\wedge R(x,y)\wedge R(y,z)\big)\vee\big(P(x)\wedge P(y)\wedge P(z)\wedge C(x,y)\wedge C(y,z)\big)$$

$$\rightsquigarrow$$

$$\sum_{x|P(x)}\Big(\tfrac{1}{8}+\sum_{y|P(y)\wedge R(x,y)}\big(\tfrac{1}{4}+\sum_{z|P(z)\wedge R(y,z)}1\big)\Big)+\sum_{x|P(x)}\Big(\tfrac{1}{8}+\sum_{y|P(y)\wedge C(x,y)}\big(\tfrac{1}{4}+\sum_{z|P(z)\wedge C(y,z)}1\big)\Big)$$

Results vs. Fluxplayer

	Toss Wins	Fluxplayer Wins	Tie
Breakthrough	95%	5%	0%
Connect4	20%	75%	5%
Connect5	0%	0%	100%
Pawn Whopping	50%	50%	0%

(fixed depth)

Deriving Heuristics

Methods

- Goal expansion and Type Normal Form
- Summing over conjuncts in existentially quantified conjunctions
- Retain stable guards and include rule preconditions

Example: Tic-Tac-Toe without Diagonals

$$\big(P(x) \wedge P(y) \wedge P(z) \wedge R(x,y) \wedge R(y,z)\big) \vee \big(P(x) \wedge P(y) \wedge P(z) \wedge C(x,y) \wedge C(y,z)\big)$$

$$\rightsquigarrow$$

$$\sum_{x|P(x)} \Big(\tfrac{1}{8} + \sum_{y|P(y)\wedge R(x,y)} \big(\tfrac{1}{4} + \sum_{z|P(z)\wedge R(y,z)} 1\big)\Big) + \sum_{x|P(x)} \Big(\tfrac{1}{8} + \sum_{y|P(y)\wedge C(x,y)} \big(\tfrac{1}{4} + \sum_{z|P(z)\wedge C(y,z)} 1\big)\Big)$$

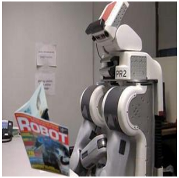
Results vs. Fluxplayer

(variable depth)

	Toss Wins	Fluxplayer Wins	Tie
Breakthrough	95%	5%	0%
Connect4	45%	20%	35%
Connect5	0%	0%	100%
Pawn Whopping	60%	40%	0%

Outlook

We can demonstrate[(i)] any[(ii)] board game and play it well[(iii)]


→


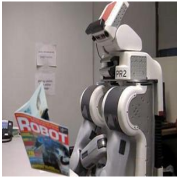
→

→


(i) Good screenshots needed, only simple constraints derived
(ii) Only perfect information games done, slow for complex games
(iii) Deeper abstraction and concept learning missing

Outlook

We can* demonstrate(i) any(ii) board game and play it well(iii)

 → 

→  → 

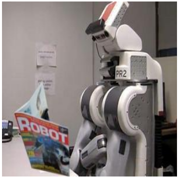
(i) Good screenshots needed, only simple constraints derived
(ii) Only perfect information games done, slow for complex games
(iii) Deeper abstraction and concept learning missing

* **but a lot is still to be done**

- Optimize for a robot. **We are looking for collaborators!**
- Adapt to **imperfect information** games and **continuous dynamics**.
- Investigate **more robust** solution methods, more games and problems.

Outlook

We can* demonstrate[i] any[ii] board game and play it well[iii]

 → 

→  → 

(i) Good screenshots needed, only simple constraints derived
(ii) Only perfect information games done, slow for complex games
(iii) Deeper abstraction and concept learning missing

* **but a lot is still to be done**

- Optimize for a robot. **We are looking for collaborators!**
- Adapt to **imperfect information** games and **continuous dynamics**.
- Investigate **more robust** solution methods, more games and problems.

Thank You (`www.toss.sf.net`)