First-order logic with game-theoretic recursion

Reijo Jaakkola
reijo.jaakkola@tuni.fi

Tampere University
Joint work with Antti Kuusisto

Funding: Theory of computational logics
Academy of Finland grants 324435 and 328987

Computational logic CL

- CL was introduced in (Kuusisto, 14), where it was also proved that it characterises the class Σ^0_1, i.e., the class of recursively enumerable languages.

Reijo Jaakkola
reijo.jaakkola@tuni.fi

Computational logic CL

- CL was introduced in (Kuusisto, 14), where it was also proved that it characterises the class Σ_1^0, i.e., the class of recursively enumerable languages.
- CL extends standard first-order logic FO with two novel features.

Reijo Jaakkola
reijo.jaakkola@tuni.fi

Computational logic CL

- CL was introduced in (Kuusisto, 14), where it was also proved that it characterises the class Σ_1^0, i.e., the class of recursively enumerable languages.
- CL extends standard first-order logic FO with two novel features.
  1. The ability to modify the underlying model: adding new elements to the domain of the model, new tuples to relations and even new relations.

Reijo Jaakkola
reijo.jaakkola@tuni.fi

Computational logic CL

- CL was introduced in (Kuusisto, 14), where it was also proved that it characterises the class Σ_1^0, i.e., the class of recursively enumerable languages.
- CL extends standard first-order logic FO with two novel features.
  1. The ability to modify the underlying model: adding new elements to the domain of the model, new tuples to relations and even new relations.
  2. The ability to use recursion (looping) via self-reference.

Computational logic CL

- CL was introduced in (Kuusisto, 14), where it was also proved that it characterises the class Σ_1^0, i.e., the class of recursively enumerable languages.
- CL extends standard first-order logic FO with two novel features.
  1. The ability to modify the underlying model: adding new elements to the domain of the model, new tuples to relations and even new relations.
  2. The ability to use recursion (looping) via self-reference.

  FO extended with just recursion (2.) is called the **static** CL (SCL).

Reijo Jaakkola
reijo.jaakkola@tuni.fi

Computational logic CL

- CL was introduced in (Kuusisto, 14), where it was also proved that it characterises the class Σ_1^0, i.e., the class of recursively enumerable languages.
- CL extends standard first-order logic FO with two novel features.

  1. The ability to modify the underlying model: adding new elements to the domain of the model, new tuples to relations and even new relations.
  2. The ability to use recursion (looping) via self-reference.

  FO extended with just recursion (2.) is called the **static** CL (SCL).
- SCL is in itself a very natural extension of FO and it bears some resembles with the programming language IND introduced in (Harel & Kozen, 84).

Reijo Jaakkola
reijo.jaakkola@tuni.fi

Computational logic CL

Reijo Jaakkola
reijo.jaakkola@tuni.fi

- CL was introduced in (Kuusisto, 14), where it was also proved that it characterises the class Σ_1^0, i.e., the class of recursively enumerable languages.
- CL extends standard first-order logic FO with two novel features.
  1. The ability to modify the underlying model: adding new elements to the domain of the model, new tuples to relations and even new relations.
  2. The ability to use recursion (looping) via self-reference.

  FO extended with just recursion (2.) is called the **static** CL (SCL).
- SCL is in itself a very natural extension of FO and it bears some resembles with the programming language IND introduced in (Harel & Kozen, 84).
- The purpose of this presentation is to present some very recent work on the proof theory of SCL.

Syntax of SCL

Reijo Jaakkola
reijo.jaakkola@tuni.fi

- We fix a set $LBS = \{L_i \mid i \in \mathbb{N}\}$ of **label symbols**.

Syntax of SCL

- We fix a set $LBS = \{L_i \mid i \in \mathbb{N}\}$ of **label symbols**.
- For each $L \in LBS$ we have a corresponding **reference symbol** C_L.

Reijo Jaakkola
reijo.jaakkola@tuni.fi

Syntax of SCL

- We fix a set $LBS = \{L_i \mid i \in \mathbb{N}\}$ of **label symbols**.
- For each $L \in LBS$ we have a corresponding **reference symbol** C_L.

Definition

Let τ be a relational vocabulary.

Reijo Jaakkola
reijo.jaakkola@tuni.fi

Syntax of SCL

- We fix a set $LBS = \{L_i \mid i \in \mathbb{N}\}$ of **label symbols**.
- For each $L \in LBS$ we have a corresponding **reference symbol** C_L.

Definition

Let τ be a relational vocabulary. The set of formulas $\mathrm{SCL}[\tau]$ is defined by the following grammar:

$$\varphi ::= R(\overline{x}) \mid C_L \mid \neg\varphi \mid \varphi \wedge \varphi \mid \exists x \varphi \mid L\varphi,$$

where $R \in \tau$ and $L \in LBS$.

Game-theoretical semantics (GTS) for SCL

- We associate to each structure $\mathcal{A}$, assignment s and a formula φ of SCL a two-player game $\mathcal{G}_\infty(\mathcal{A}, s, \varphi)$, which is essentially a reachability game.

Game-theoretical semantics (GTS) for SCL

- We associate to each structure $\mathcal{A}$, assignment s and a formula φ of SCL a two-player game $\mathcal{G}_\infty(\mathcal{A}, s, \varphi)$, which is essentially a reachability game.
- **Important:** neither player wins infinite plays.

Game-theoretical semantics (GTS) for SCL

- We associate to each structure $\mathcal{A}$, assignment s and a formula φ of SCL a two-player game $\mathcal{G}_\infty(\mathcal{A}, s, \varphi)$, which is essentially a reachability game.
- **Important:** neither player wins infinite plays.
- Positions of the game are triples $(r, \psi, \#)$, where r is the **current** assignment, ψ is a **subformula** of φ and $\# \in \{+, -\}$.

Game-theoretical semantics (GTS) for SCL

- We associate to each structure $\mathcal{A}$, assignment s and a formula φ of SCL a two-player game $\mathcal{G}_\infty(\mathcal{A}, s, \varphi)$, which is essentially a reachability game.
- **Important:** neither player wins infinite plays.
- Positions of the game are triples $(r, \psi, \#)$, where r is the **current** assignment, ψ is a **subformula** of φ and $\# \in \{+, -\}$.

$\mathcal{A}, s \models \varphi \Leftrightarrow$ Verifier has a winning strategy in the game $\mathcal{G}_\infty(\mathcal{A}, s, \varphi)$.

Game-theoretical semantics (GTS) for SCL

- Rules of the game $\mathcal{G}_\infty(\mathcal{A}, s, \varphi)$ are very natural.

Game-theoretical semantics (GTS) for SCL

- Rules of the game $\mathcal{G}_\infty(\mathcal{A}, s, \varphi)$ are very natural.
    - Game starts from the position $(s, \varphi, +)$.

Reijo Jaakkola
reijo.jaakkola@tuni.fi

Game-theoretical semantics (GTS) for SCL

- Rules of the game $\mathcal{G}_\infty(\mathcal{A}, s, \varphi)$ are very natural.
  - Game starts from the position $(s, \varphi, +)$.
  - First-order connectives are standard. E.g. in a position $(r, \neg\psi, +)$ the game continues from the position $(r, \psi, -)$.

Game-theoretical semantics (GTS) for SCL

- Rules of the game $\mathcal{G}_\infty(\mathcal{A}, s, \varphi)$ are very natural.
  - Game starts from the position $(s, \varphi, +)$.
  - First-order connectives are standard. E.g. in a position $(r, \neg\psi, +)$ the game continues from the position $(r, \psi, -)$.
  - From $(r, L\psi, \#)$ the game proceeds to $(r, \psi, \#)$.

Game-theoretical semantics (GTS) for SCL

- Rules of the game $\mathcal{G}_\infty(\mathcal{A}, s, \varphi)$ are very natural.
  - Game starts from the position $(s, \varphi, +)$.
  - First-order connectives are standard. E.g. in a position $(r, \neg\psi, +)$ the game continues from the position $(r, \psi, -)$.
  - From $(r, L\psi, \#)$ the game proceeds to $(r, \psi, \#)$.
  - From $(r, C_L, \#)$ the game can continue in two different ways.

Game-theoretical semantics (GTS) for SCL

Reijo Jaakkola
reijo.jaakkola@tuni.fi

- Rules of the game $\mathcal{G}_\infty(\mathcal{A}, s, \varphi)$ are very natural.
  - Game starts from the position $(s, \varphi, +)$.
  - First-order connectives are standard. E.g. in a position $(r, \neg\psi, +)$ the game continues from the position $(r, \psi, -)$.
  - From $(r, L\psi, \#)$ the game proceeds to $(r, \psi, \#)$.
  - From $(r, C_L, \#)$ the game can continue in two different ways.
    1. If C_L does not refer to a subformula of φ, then the game stops and neither player wins.

Game-theoretical semantics (GTS) for SCL

- Rules of the game $\mathcal{G}_\infty(\mathcal{A}, s, \varphi)$ are very natural.
  - Game starts from the position $(s, \varphi, +)$.
  - First-order connectives are standard. E.g. in a position $(r, \neg\psi, +)$ the game continues from the position $(r, \psi, -)$.
  - From $(r, L\psi, \#)$ the game proceeds to $(r, \psi, \#)$.
  - From $(r, C_L, \#)$ the game can continue in two different ways.
    1. If C_L does not refer to a subformula of φ, then the game stops and neither player wins.
    2. If C_L refers to a subformula ψ of φ, then the game proceeds to position $(r, \psi, \#)$.

Examples

Example

1. The formula C_L is undetermined in every structure.

Reijo Jaakkola
reijo.jaakkola@tuni.fi

Examples

Example

1. The formula C_L is undetermined in every structure.
2. The sentence

$$\forall x \forall y(x = y \vee L(Exy \vee \exists z(Ezy \wedge \exists y(y = z \wedge C_L))))$$

expresses that a graph is connected.

Reijo Jaakkola
reijo.jaakkola@tuni.fi

Examples

Example

1. The formula C_L is undetermined in every structure.
2. The sentence
$$\forall x \forall y (x = y \vee L(Exy \vee \exists z(Ezy \wedge \exists y(y = z \wedge C_L))))$$
expresses that a graph is connected.
3. The sentence
$$\neg \exists x L \exists y (y < x \wedge \exists x (x = y \wedge C_L))$$
expresses that $<$ is well-founded.

Complexity of SCL

- Satisfiability problem for SCL is very hard...

Reijo Jaakkola
reijo.jaakkola@tuni.fi

Complexity of SCL

- Satisfiability problem for SCL is very hard...
- On the positive side, the validity problem for SCL is in Σ^0_1.

Reijo Jaakkola
reijo.jaakkola@tuni.fi

Complexity of SCL

- Satisfiability problem for SCL is very hard...
- On the positive side, the validity problem for SCL is in Σ^0_1.
- We were able to design a proof system $\mathcal{S}$ with the following property: for every set Σ of FO-formulas and an SCL formula φ we have that

$$\Sigma \models \varphi \Leftrightarrow \Sigma \vdash_{\mathcal{S}} \varphi.$$

Complexity of SCL

- Satisfiability problem for SCL is very hard...
- On the positive side, the validity problem for SCL is in Σ_1^0.
- We were able to design a proof system $\mathcal{S}$ with the following property: for every set Σ of FO-formulas and an SCL formula φ we have that

$$\Sigma \vDash \varphi \Leftrightarrow \Sigma \vdash_{\mathcal{S}} \varphi.$$

- Main technical tool are FO-formulas that "approximate" SCL formulas.

Reijo Jaakkola
reijo.jaakkola@tuni.fi

Approximants

- For every $n \in \mathbb{N}$ the game $\mathcal{G}_n(\mathcal{A}, s, \varphi)$ is obtained from $\mathcal{G}_\infty(\mathcal{A}, s, \varphi)$ by requiring that looping can happen at most n-times.

Approximants

- For every $n \in \mathbb{N}$ the game $\mathcal{G}_n(\mathcal{A}, s, \varphi)$ is obtained from $\mathcal{G}_\infty(\mathcal{A}, s, \varphi)$ by requiring that looping can happen at most n-times.
- For every $n \in \mathbb{N}$ the nth approximant Φ^n_φ of φ describes the game $\mathcal{G}_n$.

Approximants

- For every $n \in \mathbb{N}$ the game $\mathcal{G}_n(\mathcal{A}, s, \varphi)$ is obtained from $\mathcal{G}_\infty(\mathcal{A}, s, \varphi)$ by requiring that looping can happen at most n-times.
- For every $n \in \mathbb{N}$ the nth approximant Φ_φ^n of φ describes the game $\mathcal{G}_n$.
- Φ_φ^n is a first-order formula!

Approximants

- For every $n \in \mathbb{N}$ the game $\mathcal{G}_n(\mathcal{A}, s, \varphi)$ is obtained from $\mathcal{G}_\infty(\mathcal{A}, s, \varphi)$ by requiring that looping can happen at most n-times.
- For every $n \in \mathbb{N}$ the nth approximant Φ^n_φ of φ describes the game $\mathcal{G}_n$.
- Φ^n_φ is a first-order formula!

Proposition

Verifier has a winning strategy in $\mathcal{G}_n(\mathcal{A}, s, \varphi)$ if and only if $\mathcal{A}, s \models \Phi^n_\varphi$.

Approximants and validity of SCL formulas

Reijo Jaakkola
reijo.jaakkola@tuni.fi

Theorem

Let φ be a sentence of SCL. Suppose that for every $n \in \mathbb{N}$ there exists a structure $\mathcal{A}_n$ such that Verifier does not have a winning strategy in the game $\mathcal{G}_n(\mathcal{A}_n, \varphi)$. Then there exists a structure $\mathcal{A}$ such that Verifier does not have a winning strategy in the game $\mathcal{G}_\infty(\mathcal{A}, \varphi)$.

Approximants and validity of SCL formulas

Reijo Jaakkola
reijo.jaakkola@tuni.fi

Theorem

Let φ be a sentence of SCL. Suppose that for every $n \in \mathbb{N}$ there exists a structure $\mathcal{A}_n$ such that Verifier does not have a winning strategy in the game $\mathcal{G}_n(\mathcal{A}_n, \varphi)$. Then there exists a structure $\mathcal{A}$ such that Verifier does not have a winning strategy in the game $\mathcal{G}_\infty(\mathcal{A}, \varphi)$.

Corollary

A sentence of SCL is valid if and only if one of its approximants is.

Very rough sketch of the proof system

- Make the proof system FO-complete so that we can deduce all the valid approximants.

Reijo Jaakkola
reijo.jaakkola@tuni.fi

Very rough sketch of the proof system

- Make the proof system FO-complete so that we can deduce all the valid approximants.
- Add enough rules so that we can deduce from approximants the corresponding SCL sentences.

Reijo Jaakkola
reijo.jaakkola@tuni.fi

Very rough sketch of the proof system

- Make the proof system FO-complete so that we can deduce all the valid approximants.
- Add enough rules so that we can deduce from approximants the corresponding SCL sentences.
- **Unfortunately not so straightforward...**

Some open problems and future directions

- What is the complexity of the satisfiability problem of SCL^2? In particular, is it decidable?

Some open problems and future directions

- What is the complexity of the satisfiability problem of SCL^2? In particular, is it decidable?
- Design a useful model comparison game (or EF-game) for SCL.

Reijo Jaakkola
reijo.jaakkola@tuni.fi

Some open problems and future directions

- What is the complexity of the satisfiability problem of SCL^2? In particular, is it decidable?
- Design a useful model comparison game (or EF-game) for SCL.
- Developing simpler weakly complete proof systems for SCL.

Reijo Jaakkola
reijo.jaakkola@tuni.fi

Thanks for listening! :) Questions?