FWF
Der Wissenschaftsfonds.

EPSRC
Engineering and Physical Sciences Research Council

Reasoning with Forest Logic Programs Using Fully Enriched Automata

Cristina Feier[1] Thomas Eiter[2]
presenter: Mantas Šimkus[2]

[1] FB 03, University of Bremen, Bremen Germany
[2] Institute of Information Systems, Vienna University of Technology, Vienna Austria

13th International Conference on
Logic Programming and Nonmonotonic Reasoning
Lexington, Kentucky, US

Acknowledgement: This work is partially supported by the EPSRC grants Score! and DBOnto and the Austrian Science Fund (FWF) grants P24090 and P25207

FWF Der Wissenschaftsfonds.

Introduction

EPSRC Engineering and Physical Sciences Research Council

Forest Logic Programs:

- decidable fragment of Open Answer Set Programming
- non-monotonic language and rule-based syntax
- open domain semantics
- can simulate reasoning with the expressive DL $\mathcal{SHOQ}$

Previous work:

- non-deterministic tableau algorithms: 2NEXPTIME, NEXPTIME running time
- exact complexity characterization still open

Current work:

- encoding of reasoning with FoLPs into emptiness checking of fully enriched automata $\Longrightarrow$ EXPTIME procedure $\Longrightarrow$ worst-case optimal

Closed vs. Open World Reasoning

$$\begin{aligned} fail(X) &\leftarrow not\ pass(X) \\ pass(john) &\leftarrow \end{aligned}$$

$\rightarrow$ *ground* the program with all constants ($john$):

$$\begin{aligned} fail(john) &\leftarrow not\ pass(john) \\ pass(john) &\leftarrow \end{aligned}$$

$\rightarrow$ answer set: $\{pass(john)\}$.
$\rightarrow$ *fail* is not satisfiable:

- assume the presence of anonymous objects – *open domains*
- e.g. with universe $\{john, x\}$, *fail* becomes satisfiable

FWF Der Wissenschaftsfonds.

Open Answer Set Programming

EPSRC Engineering and Physical Sciences Research Council

Enhancing Answer Set Programming with open domains:

Syntax

same as the syntax of *function-free* Answer Set Programming

Semantics (OASP)

- (U, M) is an *open answer set* of an OASP (FoLP) P, iff $U \supseteq cts(P)$ and M is an answer set of P_U

When $U = \{john, x\}$, P_U:

$$\begin{aligned} fail(john) &\leftarrow not\ pass(john) \\ fail(x) &\leftarrow not\ pass(x) \\ pass(john) &\leftarrow \end{aligned}$$

$M = \{pass(john), fail(x)\}$ is an answer set of P_U: $\rightsquigarrow$
$(\{john, x\}, \{pass(john), fail(x)\})$ is an open answer set!

Forest Logic Programs

OASP is undecidable: syntactical restrictions to achieve decidability;

Forest Logic Programs

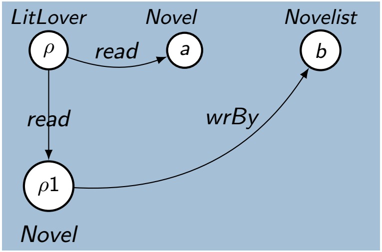
- allow only for unary and binary predicates
- tree-shaped rules: *forest model property*
- a special type of unsafe rules: *free rules*
- facts

$$
\begin{aligned}
r_1 &: & LitLover(X) &\leftarrow read(X, Y_1), read(X, Y_2), \\
& & & \quad Novel(Y_1), Novel(Y_2), Y_1 \neq Y_2 \\
r_2 &: & Novel(X) &\leftarrow wrBy(X, Y), Novelist(Y) \\
r_3 &: & Novelist(X) &\leftarrow wrote(X, Y), Novel(Y) \\
r_4 &: & read(X, Y) \vee not\ read(X, Y) &\leftarrow \\
r_5 &: & wrBy(X, Y) \vee not\ wrBy(X, Y) &\leftarrow \\
r_6 &: & wrote(X, Y) \vee not\ wrote(X, Y) &\leftarrow \\
f_1 &: & Novel(a) &\leftarrow \\
f_2 &: & Novelist(b) &\leftarrow
\end{aligned}
$$

Forest model property

A unary predicate is satisfiable iff it is satisfied by a forest-shaped model

$r_1 : LitLover(X) \leftarrow read(X, Y_1), read(X, Y_2),$
$\qquad Novel(Y_1), Novel(Y_2), Y_1 \neq Y_2.$
$r_2 : Novel(X) \leftarrow wrBy(X, Y), Novelist(Y).$
$r_3 : Novelist(X) \leftarrow wrote(X, Y), Novel(Y).$
$r_4 : read(X, Y) \vee not\ read(X, Y) \leftarrow .$
$r_5 : wrBy(X, Y) \vee not\ wrBy(X, Y) \leftarrow .$
$r_6 : wrote(X, Y) \vee not\ wrote(X, Y) \leftarrow .$
$f_1 : Novel(a).$
$f_2 : Novelist(b).$



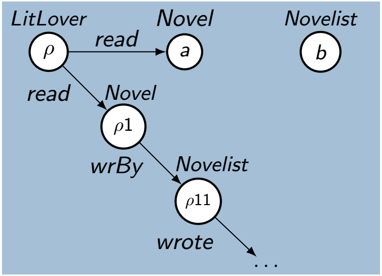
(U, M) with:

- $U = \{\rho, \rho 1, a, b\}$, and
- $M = \{LitLover(\rho), Novel(a), read(\rho, \rho 1), \ldots\}$

is a forest model which satisfies *LitLover*

Forest Models are Well-Supported

$r_1 : LitLover(X) \leftarrow read(X, Y_1), read(X, Y_2), Novel(Y_1), Novel(Y_2), Y_1 \neq Y_2.$
$r_2 : Novel(X) \leftarrow wrBy(X, Y), Novelist(Y).$
$r_3 : Novelist(X) \leftarrow wrote(X, Y), Novel(Y).$
$r_4 : read(X, Y) \vee not\ read(X, Y) \leftarrow .$
$r_5 : wrBy(X, Y) \vee not\ wrBy(X, Y) \leftarrow .$
$r_6 : wrote(X, Y) \vee not\ wrote(X, Y) \leftarrow .$
$f_1 : Novel(a).$
$f_2 : Novelist(b).$



(U, M) with:

- $U = \{\rho, \rho 1, \ldots, a, b\}$, and
- $M = \{LitLover(\rho), Novel(a), Novelist(b)\ read(\rho, \rho 1), Novel(\rho 1), \ldots\}$

is not a forest model!

Constructing well-supported models

Done in the past using tableaux algorithms:

- blocking mechanism incorporates a well-supportedness check
- usually non-deterministic: 2NExpTime, NExpTime running times
- worst-case optimal (ExpTime) AND/OR tableaux algorithm devised for the case of CoLPs (FoLPs$\setminus$constants)
- AND/OR technique does not generalize to FoLPs
- complexity gap: satisfiability checking w.r.t. FoLPs was known to be ExpTime-hard

Fully Enriched Automata

- Run on labeled forests
- Introduced as a device to reason with hybrid graded μ-calculus

$A = \langle \Sigma, b, Q, \delta, q_0, \mathcal{F} \rangle$:

- Σ is a finite input alphabet
- $b > 0$ is a counting bound
- Q is a finite set of states
- $\delta : Q \times \Sigma \to B^+(D_b \times Q)$ - the transition function, where:
  - $B^+(Y)$ is the set of positive Boolean formulas over Y
  - $D_b = \{\langle 0\rangle, \langle 1\rangle, \ldots, \langle b\rangle\} \cup \{[0], [1], \ldots, [b]\} \cup \{-1, \varepsilon, \langle root\rangle, [root]\}$
- $q_0 \in Q$ - the initial state
- $\mathcal{F} = \{\mathcal{F}_1, \mathcal{F}_2, \ldots, \mathcal{F}_k\}$, where $\mathcal{F}_1 \subseteq \mathcal{F}_2 \subseteq \ldots \subseteq \mathcal{F}_k = Q$ is a *parity acceptance condition*

Emptiness checking for a FEA A as above with n states can be decided in time $(b+2)^{\mathcal{O}(n^3 \cdot k^2 \cdot \log k \cdot \log b^2)}$.

Reasoning with FoLPs Using FEA

For every FoLP P and unary predicate p construct a class of FEA $A^{p,P}_{\rho,\theta}$:

- ρ is a designated constant or anonymous node
- θ fixes a label for each root node of accepted forests
- states of the form $q_{t,a}$, q_{t,r_a}, etc. where t is a term pattern (a designated constant or *), a is a unary predicate, r_a is a unary rule, etc.
- number of states: polynomial in the size of P
- parity acceptance condition: $\mathcal{F} = (\mathcal{F}_1, \mathcal{F}_2)$
  - $\mathcal{F}_1 = \{q_{t,a}, q_{t_1,t_2,f} \mid a/f$ a unary/binary predicate; t, t_1 and t_2 term patterns $\}$,
  - $\mathcal{F}_2 = Q$
  - exploited for checking well-supportedness

For details about the encoding, please check the paper!

Properties of the Encoding

For a FoLP P and a unary predicate symbol p, p is satisfiable w.r.t. P iff there exists an automaton $A^{p,P}_{\rho,\theta}$ whose language is non-empty.

Satisfiability checking of unary predicates with respect to FoLPs is EXPTIME-complete.

f-hybrid KBs: pairs (Σ, P)

- Σ a $\mathcal{SHOQ}$ kb, P a FoLP: no restriction on signature sharing
- a unary predicate p is satisfiable w.r.t. (Σ, P) iff it is satisfiable w.r.t. $\Theta(\Sigma) \cup P$, where Θ is a polynomial and modular translation from $\mathcal{SHOQ}$ to FoLPs.

Satisfiability checking of unary predicates with respect to f-hybrid KBs is EXPTIME-complete.

Conclusions

The result closes an open problem: exact complexity characterization of FoLPs

FEAs - elegant device for encoding

- accept forests as input
- parity acceptance condition to check well-supportedness
- additional addressing and term matching mechanisms needed

Existing work on AND/OR tableau reasoners for CoLPs (FoLPs minus constants):

- how can it be lifted to FoLPs?

Questions?