Exploiting forwardness: Satisfiability and Query Entailment in

Forward Guarded Fragment

May 17, 2021, JELIA 2021

Bartosz "Bart" Bednarczyk

TU Dresden & University of Wrocław

TECHNISCHE UNIVERSITÄT DRESDEN

Uniwersytet Wrocławski

DeciGUT

erc
European Research Council
Established by the European Commission

Our motivation: what features make CQ answering hard for $\mathcal{ALC}$?

Our motivation: what features make CQ answering hard for $\mathcal{ALC}$?

1. Some of them behaves nice, e.g. $\mathcal{ALC}$, $\mathcal{ALC}+\mathcal{H}$, $\mathcal{ALC}+\mathcal{Q}$ [Lutz'08]

Our motivation: what features make CQ answering hard for $\mathcal{ALC}$?

1. Some of them behaves nice, e.g. $\mathcal{ALC}$, $\mathcal{ALC}+\mathcal{H}$, $\mathcal{ALC}+\mathcal{Q}$ [Lutz'08]

$\texttt{hasMother} \subseteq \texttt{hasParent}$

Our motivation: what features make CQ answering hard for $\mathcal{ALC}$?

1. Some of them behaves nice, e.g. $\mathcal{ALC}$, $\mathcal{ALC}+\mathcal{H}$, $\mathcal{ALC}+\mathcal{Q}$ [Lutz'08]

$\mathtt{hasMother} \subseteq \mathtt{hasParent}$

$\mathit{Car} \sqsubseteq (=4).\mathtt{hasPart}\,\mathit{Wheel}$

Our motivation: what features make CQ answering hard for $\mathcal{ALC}$?

1. Some of them behaves nice, e.g. $\mathcal{ALC}$, $\mathcal{ALC}+\mathcal{H}$, $\mathcal{ALC}+\mathcal{Q}$ [Lutz'08]

$\texttt{hasMother} \subseteq \texttt{hasParent}$

$\mathit{Car} \sqsubseteq (=4).\texttt{hasPart}\,\mathit{Wheel}$

Also with arithmetic and statistical properties [Baader, B., Rudolph'20]

Our motivation: what features make CQ answering hard for $\mathcal{ALC}$?

1. Some of them behaves nice, e.g. $\mathcal{ALC}$, $\mathcal{ALC}+\mathcal{H}$, $\mathcal{ALC}+\mathcal{Q}$ [Lutz'08]

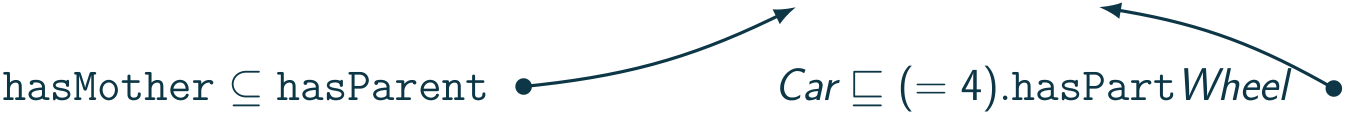
$\texttt{hasMother} \subseteq \texttt{hasParent}$ $\quad$ $\mathit{Car} \sqsubseteq (=4).\texttt{hasPart}\,\mathit{Wheel}$

Also with arithmetic and statistical properties [Baader, B., Rudolph'20]

As well as with regular expr, fixed points, (safe) role combination [B.'21, in prep.]

Our motivation: what features make CQ answering hard for $\mathcal{ALC}$?

1. Some of them behaves nice, e.g. $\mathcal{ALC}$, $\mathcal{ALC}+\mathcal{H}$, $\mathcal{ALC}+\mathcal{Q}$ [Lutz'08]



$\mathtt{hasMother} \subseteq \mathtt{hasParent}$

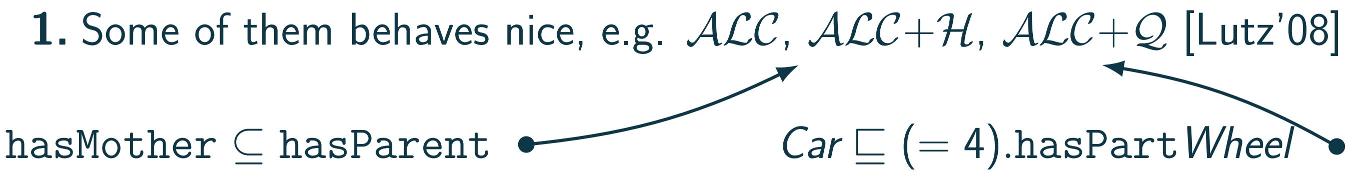
$\mathit{Car} \sqsubseteq (= 4).\mathtt{hasPart}\,\mathit{Wheel}$

Also with arithmetic and statistical properties [Baader, B., Rudolph'20]

As well as with regular expr, fixed points, (safe) role combination [B.'21, in prep.]

2. Some of them increase the complexity exponentially:

Our motivation: what features make CQ answering hard for $\mathcal{ALC}$?



1. Some of them behaves nice, e.g. $\mathcal{ALC}$, $\mathcal{ALC}+\mathcal{H}$, $\mathcal{ALC}+\mathcal{Q}$ [Lutz'08]

$\texttt{hasMother} \subseteq \texttt{hasParent}$

$\mathit{Car} \sqsubseteq (=4).\texttt{hasPart}\,\mathit{Wheel}$

Also with arithmetic and statistical properties [Baader, B., Rudolph'20]

As well as with regular expr, fixed points, (safe) role combination [B.'21, in prep.]

2. Some of them increase the complexity exponentially:

E.g. transitivity [Eiter et al.'09], nominals (a.k.a. constants) [Ngo et al.'16]

Our motivation: what features make CQ answering hard for $\mathcal{ALC}$?

1. Some of them behaves nice, e.g. $\mathcal{ALC}$, $\mathcal{ALC}+\mathcal{H}$, $\mathcal{ALC}+\mathcal{Q}$ [Lutz'08]

$\texttt{hasMother} \subseteq \texttt{hasParent}$ $\qquad$ $\mathit{Car} \sqsubseteq (=4).\texttt{hasPart}\,\mathit{Wheel}$

Also with arithmetic and statistical properties [Baader, B., Rudolph'20]

As well as with regular expr, fixed points, (safe) role combination [B.'21, in prep.]

2. Some of them increase the complexity exponentially:

E.g. transitivity [Eiter et al.'09], nominals (a.k.a. constants) [Ngo et al.'16]

more: inverses [Lutz'07], self-loops [B., Rudolph'21 Submitted.]

Our motivation: what features make CQ answering hard for $\mathcal{ALC}$?

1. Some of them behaves nice, e.g. $\mathcal{ALC}$, $\mathcal{ALC}{+}\mathcal{H}$, $\mathcal{ALC}{+}\mathcal{Q}$ [Lutz'08]

$\mathtt{hasMother} \subseteq \mathtt{hasParent}$ $\qquad$ $\mathit{Car} \sqsubseteq (= 4).\mathtt{hasPart}\,\mathit{Wheel}$

Also with arithmetic and statistical properties [Baader, B., Rudolph'20]

As well as with regular expr, fixed points, (safe) role combination [B.'21, in prep.]

2. Some of them increase the complexity exponentially:

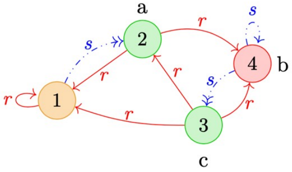
E.g. transitivity [Eiter et al.'09], nominals (a.k.a. constants) [Ngo et al.'16]

more: inverses [Lutz'07], self-loops [B., Rudolph'21 Submitted.]

What makes $\mathcal{ALC}$ easy, but $\mathcal{ALCI}$ and the others hard?

Our motivation: ng hard for $\mathcal{ALC}$?

1. Some of them beh '$+\mathcal{Q}$ [Lutz'08]

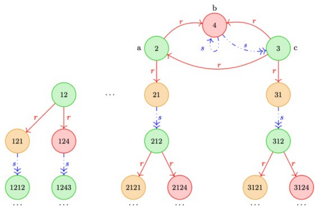


`hasMother` $\subseteq$ `hasPar` `Part`*Wheel*

Also with arithmetic ar udolph'20]

As well as with regular ation [B.'21, in prep.]

2. Some of them incr



E.g. transitivity [Eiter [Ngo et al.'16]

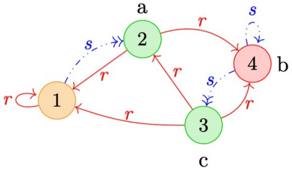
more: inverses [Lutz'07 ed.]

What makes $\mathcal{ALC}$ easy, but $\mathcal{ALCI}$ and the others hard?

Answer: Forward models!

Our motivation: ng hard for $\mathcal{ALC}$?

1. Some of them beh $+\mathcal{Q}$ [Lutz'08]

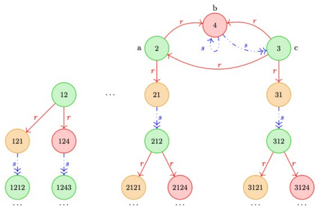


`hasMother` $\subseteq$ `hasPar` `Part`*Wheel*

Also with arithmetic ar udolph'20]

As well as with regular ation [B.'21, in prep.]

2. Some of them incr



E.g. transitivity [Eiter [Ngo et al.'16]

more: inverses [Lutz'07 ed.]

What makes $\mathcal{ALC}$ easy, but $\mathcal{ALCI}$ and the others hard?

Answer: Forward models!

Can we find a higher-arity version of $\mathcal{ALC}$ with ExpTime querying?

Our motivation: ng hard for $\mathcal{ALC}$?

1. Some of them beh '+$\mathcal{Q}$ [Lutz'08]

`hasMother` $\subseteq$ `hasPar` `Part`*Wheel*

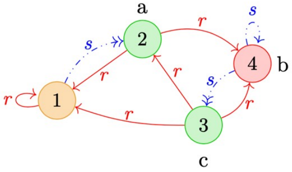
Also with arithmetic ar udolph'20]

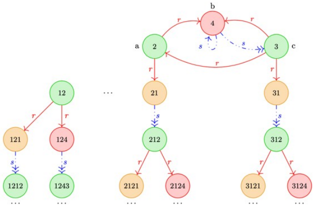
As well as with regular ation [B.'21, in prep.]

2. Some of them incr

E.g. transitivity [Eiter e [Ngo et al.'16]

more: inverses [Lutz'07 ed.]





What makes $\mathcal{ALC}$ easy, but $\mathcal{ALCI}$ and the others hard?

Answer: Forward models!

Can we find a higher-arity version of $\mathcal{ALC}$ with ExpTime querying?

Yes! $\mathcal{FGF}$ [B. JELIA'21, This talk!]

Two nice logics: $\mathcal{GF}$ [Andreka et al. 1998] and $\mathcal{FL}$ [Quine 1969]

Two nice logics: $\mathcal{GF}$ [Andreka et al. 1998] and $\mathcal{FL}$ [Quine 1969]

- The guarded fragment of $\mathcal{FO}$ is obtained by relativising quantifiers by atoms.

Two nice logics: $\mathcal{GF}$ [Andreka et al. 1998] and $\mathcal{FL}$ [Quine 1969]

- The guarded fragment of $\mathcal{FO}$ is obtained by relativising quantifiers by atoms.
- $\exists\vec{y}\ \alpha(\vec{x},\vec{y})\wedge\varphi(\vec{x},\vec{y}), \forall\vec{y}\ \alpha(\vec{x},\vec{y})\rightarrow\varphi(\vec{x},\vec{y})$ – guard must cover free variables of φ.

Two nice logics: $\mathcal{GF}$ [Andreka et al. 1998] and $\mathcal{FL}$ [Quine 1969]

- The guarded fragment of $\mathcal{FO}$ is obtained by relativising quantifiers by atoms.
- $\exists \vec{y}\ \alpha(\vec{x}, \vec{y}) \wedge \varphi(\vec{x}, \vec{y}), \forall \vec{y}\ \alpha(\vec{x}, \vec{y}) \rightarrow \varphi(\vec{x}, \vec{y})$ – guard must cover free variables of φ.

Example 1. Some artist admires only beekeepers

$$\exists x\ artst(x) \wedge \forall y\ (adm(x, y) \rightarrow bkpr(y))$$

Two nice logics: $\mathcal{GF}$ [Andreka et al. 1998] and $\mathcal{FL}$ [Quine 1969]

- The guarded fragment of $\mathcal{FO}$ is obtained by relativising quantifiers by atoms.
- $\exists\vec{y}\ \alpha(\vec{x},\vec{y})\wedge\varphi(\vec{x},\vec{y}), \forall\vec{y}\ \alpha(\vec{x},\vec{y})\rightarrow\varphi(\vec{x},\vec{y})$ – guard must cover free variables of φ.

Example 1. Some artist admires only beekeepers

$$\exists x\ artst(x) \wedge \forall y\ \ (adm(x,y) \rightarrow bkpr(y))$$

Example 2. Every artist envies every bekeeper he admires

$$\forall x\ artst(x) \rightarrow \forall y\ [adm(x,y) \rightarrow (bkpr(y) \rightarrow env(x,y))]$$

Two nice logics: $\mathcal{GF}$ [Andreka et al. 1998] and $\mathcal{FL}$ [Quine 1969]

- The guarded fragment of $\mathcal{FO}$ is obtained by relativising quantifiers by atoms.
- $\exists\vec{y}\ \alpha(\vec{x},\vec{y})\wedge\varphi(\vec{x},\vec{y}), \forall\vec{y}\ \alpha(\vec{x},\vec{y})\rightarrow\varphi(\vec{x},\vec{y})$ – guard must cover free variables of φ.

Example 1. Some artist admires only beekeepers

$$\exists x\ artst(x) \wedge \forall y\ \ (adm(x,y) \rightarrow bkpr(y))$$

Example 2. Every artist envies every bekeeper he admires

$$\forall x\ artst(x) \rightarrow \forall y\ [adm(x,y) \rightarrow (bkpr(y) \rightarrow env(x,y))]$$

Coexample 3. Every artist admires every beekeeper

$$\forall x\ (artst(x) \rightarrow \forall y\ (bkpr(y) \rightarrow adm(x,y)))$$

Two nice logics: $\mathcal{GF}$ [Andreka et al. 1998] and $\mathcal{FL}$ [Quine 1969]

- The guarded fragment of $\mathcal{FO}$ is obtained by relativising quantifiers by atoms.
- $\exists \vec{y}\ \alpha(\vec{x}, \vec{y}) \wedge \varphi(\vec{x}, \vec{y}), \forall \vec{y}\ \alpha(\vec{x}, \vec{y}) \rightarrow \varphi(\vec{x}, \vec{y})$ – guard must cover free variables of φ.

Example 1. Some artist admires only beekeepers

$$\exists x\ artst(x) \wedge \forall y\ \ (adm(x, y) \rightarrow bkpr(y))$$

Example 2. Every artist envies every bekeeper he admires

$$\forall x\ artst(x) \rightarrow \forall y\ [adm(x, y) \rightarrow (bkpr(y) \rightarrow env(x, y))]$$

Coexample 3. Every artist admires every beekeeper

$$\forall x\ (artst(x) \rightarrow \forall y\ (bkpr(y) \rightarrow adm(x, y)))$$

Theorem (Grädel 1999)

The satisfiability problem for $\mathcal{GF}$ is $2\textsc{ExpTime}$-complete.

Two nice logics: $\mathcal{GF}$ [Andreka et al. 1998] and $\mathcal{FL}$ [Quine 1969]

- The guarded fragment of $\mathcal{FO}$ is obtained by relativising quantifiers by atoms.
- $\exists\vec{y}\ \alpha(\vec{x},\vec{y}) \wedge \varphi(\vec{x},\vec{y}), \forall\vec{y}\ \alpha(\vec{x},\vec{y}) \rightarrow \varphi(\vec{x},\vec{y})$ – guard must cover free variables of φ.

Example 1. Some artist admires only beekeepers

$$\exists x\ artst(x) \wedge \forall y\ \ (adm(x,y) \rightarrow bkpr(y))$$

Example 2. Every artist envies every bekeeper he admires

$$\forall x\ artst(x) \rightarrow \forall y\ [adm(x,y) \rightarrow (bkpr(y) \rightarrow env(x,y))]$$

Coexample 3. Every artist admires every beekeeper

$$\forall x\ (artst(x) \rightarrow \forall y\ (bkpr(y) \rightarrow adm(x,y)))$$

Theorem (Grädel 1999)

The satisfiability problem for $\mathcal{GF}$ is $2\textsc{ExpTime}$-complete.

Theorem (Bárány et al. 2013)

Conjunctive query entailment problem for $\mathcal{GF}$ is $2\textsc{ExpTime}$-complete.

Two nice logics: $\mathcal{GF}$ [Andreka et al. 1998] and $\mathcal{FL}$ [Quine 1969]

Two nice logics: $\mathcal{GF}$ [Andreka et al. 1998] and $\mathcal{FL}$ [Quine 1969]

- The fluted fragment of $\mathcal{FO}$ is obtained by keeping the variables ordered.

Two nice logics: $\mathcal{GF}$ [Andreka et al. 1998] and $\mathcal{FL}$ [Quine 1969]

- The fluted fragment of $\mathcal{FO}$ is obtained by keeping the variables ordered.
- In atoms we can use only suffixes of the sequences of already quantified variables.

Two nice logics: $\mathcal{GF}$ [Andreka et al. 1998] and $\mathcal{FL}$ [Quine 1969]

- The fluted fragment of $\mathcal{FO}$ is obtained by keeping the variables ordered.
- In atoms we can use only suffixes of the sequences of already quantified variables.

Example 1. No student admires every professor

$$\forall x_1(stud(x_1) \rightarrow \neg\forall x_2(prof(x_2) \rightarrow admires(x_1, x_2)))$$

Two nice logics: $\mathcal{GF}$ [Andreka et al. 1998] and $\mathcal{FL}$ [Quine 1969]

- The fluted fragment of $\mathcal{FO}$ is obtained by keeping the variables ordered.
- In atoms we can use only suffixes of the sequences of already quantified variables.

Example 1. No student admires every professor

$$\forall x_1 (\mathit{stud}(x_1) \to \neg \forall x_2 (\mathit{prof}(x_2) \to \mathit{admires}(x_1, x_2)))$$

Example 2. No lecturer introduces any professor to every student

$$\forall x_1 (\mathit{lect}(x_1) \to \neg \exists x_2 (\mathit{prof}(x_2) \land \forall x_3 (\mathit{stud}(x_3) \to \mathit{intro}(x_1, x_2, x_3))))$$

Two nice logics: $\mathcal{GF}$ [Andreka et al. 1998] and $\mathcal{FL}$ [Quine 1969]

- The fluted fragment of $\mathcal{FO}$ is obtained by keeping the variables ordered.
- In atoms we can use only suffixes of the sequences of already quantified variables.

Example 1. No student admires every professor

$$\forall x_1 (stud(x_1) \rightarrow \neg \forall x_2 (prof(x_2) \rightarrow admires(x_1, x_2)))$$

Example 2. No lecturer introduces any professor to every student

$$\forall x_1 (lect(x_1) \rightarrow \neg \exists x_2 (prof(x_2) \wedge \forall x_3 (stud(x_3) \rightarrow intro(x_1, x_2, x_3))))$$

Coexample 1. $\forall x_1 r(x_1, x_1)$

Two nice logics: $\mathcal{GF}$ [Andreka et al. 1998] and $\mathcal{FL}$ [Quine 1969]

- The fluted fragment of $\mathcal{FO}$ is obtained by keeping the variables ordered.
- In atoms we can use only suffixes of the sequences of already quantified variables.

Example 1. No student admires every professor

$$\forall x_1 (stud(x_1) \rightarrow \neg \forall x_2 (prof(x_2) \rightarrow admires(x_1, x_2)))$$

Example 2. No lecturer introduces any professor to every student

$$\forall x_1 (lect(x_1) \rightarrow \neg \exists x_2 (prof(x_2) \wedge \forall x_3 (stud(x_3) \rightarrow intro(x_1, x_2, x_3))))$$

Coexample 1. $\forall x_1 r(x_1, x_1)$

Coexample 2. $\forall x_1 \forall x_2 r(x_1, x_2) \rightarrow s(x_2, x_1)$

Two nice logics: $\mathcal{GF}$ [Andreka et al. 1998] and $\mathcal{FL}$ [Quine 1969]

- The fluted fragment of $\mathcal{FO}$ is obtained by keeping the variables ordered.
- In atoms we can use only suffixes of the sequences of already quantified variables.

Example 1. No student admires every professor

$$\forall x_1 (stud(x_1) \to \neg \forall x_2 (prof(x_2) \to admires(x_1, x_2)))$$

Example 2. No lecturer introduces any professor to every student

$$\forall x_1 (lect(x_1) \to \neg \exists x_2 (prof(x_2) \wedge \forall x_3 (stud(x_3) \to intro(x_1, x_2, x_3))))$$

Coexample 1. $\forall x_1 r(x_1, x_1)$

Coexample 2. $\forall x_1 \forall x_2 r(x_1, x_2) \to s(x_2, x_1)$

Coexample 3. $\forall x_1 \forall x_2 \forall x_3 r(x_1, x_2) \wedge r(x_2, x_3) \to r(x_1, x_3)$

Two nice logics: $\mathcal{GF}$ [Andreka et al. 1998] and $\mathcal{FL}$ [Quine 1969]

- The fluted fragment of $\mathcal{FO}$ is obtained by keeping the variables ordered.
- In atoms we can use only suffixes of the sequences of already quantified variables.

Example 1. No student admires every professor

$$\forall x_1 (\mathit{stud}(x_1) \rightarrow \neg \forall x_2 (\mathit{prof}(x_2) \rightarrow \mathit{admires}(x_1, x_2)))$$

Example 2. No lecturer introduces any professor to every student

$$\forall x_1 (\mathit{lect}(x_1) \rightarrow \neg \exists x_2 (\mathit{prof}(x_2) \wedge \forall x_3 (\mathit{stud}(x_3) \rightarrow \mathit{intro}(x_1, x_2, x_3))))$$

Coexample 1. $\forall x_1 r(x_1, x_1)$

Coexample 2. $\forall x_1 \forall x_2 r(x_1, x_2) \rightarrow s(x_2, x_1)$

Coexample 3. $\forall x_1 \forall x_2 \forall x_3 r(x_1, x_2) \wedge r(x_2, x_3) \rightarrow r(x_1, x_3)$

Theorem (Pratt-Hartman et al. 2016)

The satisfiability problem for $\mathcal{FL}$ is $\textsc{Tower}$-complete.

Two nice logics: $\mathcal{GF}$ [Andreka et al. 1998] and $\mathcal{FL}$ [Quine 1969]

- The fluted fragment of $\mathcal{FO}$ is obtained by keeping the variables ordered.
- In atoms we can use only suffixes of the sequences of already quantified variables.

Example 1. No student admires every professor

$$\forall x_1 (stud(x_1) \rightarrow \neg \forall x_2 (prof(x_2) \rightarrow admires(x_1, x_2)))$$

Example 2. No lecturer introduces any professor to every student

$$\forall x_1 (lect(x_1) \rightarrow \neg \exists x_2 (prof(x_2) \land \forall x_3 (stud(x_3) \rightarrow intro(x_1, x_2, x_3))))$$

Coexample 1. $\forall x_1 r(x_1, x_1)$

Coexample 2. $\forall x_1 \forall x_2 r(x_1, x_2) \rightarrow s(x_2, x_1)$

Coexample 3. $\forall x_1 \forall x_2 \forall x_3 r(x_1, x_2) \land r(x_2, x_3) \rightarrow r(x_1, x_3)$

Theorem (Pratt-Hartman et al. 2016)

The satisfiability problem for $\mathcal{FL}$ is $\textsc{Tower}$-complete.

If we replace suffices by infixes in $\mathcal{FL}$ we get the forward fragment $\mathcal{FF}$.

Two nice logics: $\mathcal{GF}$ [Andreka et al. 1998] and $\mathcal{FL}$ [Quine 1969]

- The fluted fragment of $\mathcal{FO}$ is obtained by keeping the variables ordered.
- In atoms we can use only suffixes of the sequences of already quantified variables.

Example 1. No student admires every professor

$$\forall x_1 (stud(x_1) \rightarrow \neg \forall x_2 (prof(x_2) \rightarrow admires(x_1, x_2)))$$

Example 2. No lecturer introduces any professor to every student

$$\forall x_1 (lect(x_1) \rightarrow \neg \exists x_2 (prof(x_2) \wedge \forall x_3 (stud(x_3) \rightarrow intro(x_1, x_2, x_3))))$$

Coexample 1. $\forall x_1 r(x_1, x_1)$

Coexample 2. $\forall x_1 \forall x_2 r(x_1, x_2) \rightarrow s(x_2, x_1)$

Coexample 3. $\forall x_1 \forall x_2 \forall x_3 r(x_1, x_2) \wedge r(x_2, x_3) \rightarrow r(x_1, x_3)$

Theorem (Pratt-Hartman et al. 2016)

The satisfiability problem for $\mathcal{FL}$ is $\textsc{Tower}$-complete.

If we replace suffices by infixes in $\mathcal{FL}$ we get the forward fragment $\mathcal{FF}$.

Lemma (B. 2021)

$\mathcal{FF}$ is reducible to $\mathcal{FL}$ in polynomial time.

Two nice logics: $\mathcal{GF}$ [Andreka et al. 1998] and $\mathcal{FL}$ [Quine 1969]

Two nice logics: $\mathcal{GF}$ [Andreka et al. 1998] and $\mathcal{FL}$ [Quine 1969]

Both $\mathcal{GF}$ and $\mathcal{FF}$ capture $\mathcal{ALC}$, e.g.: “Grandfathers with granddaughters”

$$\texttt{grf-wth-gdtrs} \sqsubseteq \exists \texttt{hasChld}.\exists \texttt{hasChld}.\mathit{female}$$

Two nice logics: $\mathcal{GF}$ [Andreka et al. 1998] and $\mathcal{FL}$ [Quine 1969]

Both $\mathcal{GF}$ and $\mathcal{FF}$ capture $\mathcal{ALC}$, e.g.: “Grandfathers with granddaughters”

$$\texttt{grf-wth-gdtrs} \sqsubseteq \exists\texttt{hasChld}.\exists\texttt{hasChld}.\mathit{female}$$

In $\mathcal{GF}$:

$$\forall x\ \texttt{grf-wth-gdtrs}(x) \to \exists y\ \texttt{hasChld}(x, y) \land \exists z\ \texttt{hasChld}(y, z) \land \mathit{female}(z)$$

Two nice logics: $\mathcal{GF}$ [Andreka et al. 1998] and $\mathcal{FL}$ [Quine 1969]

Both $\mathcal{GF}$ and $\mathcal{FF}$ capture $\mathcal{ALC}$, e.g.: “Grandfathers with granddaughters”

$$\texttt{grf-wth-gdtrs} \sqsubseteq \exists \texttt{hasChld}.\exists \texttt{hasChld}.\mathit{female}$$

In $\mathcal{GF}$:

$$\forall x\ \texttt{grf-wth-gdtrs}(x) \rightarrow \exists y\ \texttt{hasChld}(x, y) \wedge \exists z\ \texttt{hasChld}(y, z) \wedge \mathit{female}(z)$$

In $\mathcal{FF}$:

$$\forall x_1\ \texttt{grf-wth-gdtrs}(x_1) \rightarrow \exists x_2\ \texttt{hasChld}(x_1, x_2) \wedge \exists x_3\ \texttt{hasChld}(x_2, x_3) \wedge \mathit{female}(x_3)$$

Two nice logics: $\mathcal{GF}$ [Andreka et al. 1998] and $\mathcal{FL}$ [Quine 1969]

Both $\mathcal{GF}$ and $\mathcal{FF}$ capture $\mathcal{ALC}$, e.g.: “Grandfathers with granddaughters”

$$\texttt{grf-wth-gdtrs} \sqsubseteq \exists\texttt{hasChld}.\exists\texttt{hasChld}.\mathit{female}$$

In $\mathcal{GF}$:

$$\forall x\ \texttt{grf-wth-gdtrs}(x) \to \exists y\ \texttt{hasChld}(x, y) \wedge \exists z\ \texttt{hasChld}(y, z) \wedge \mathit{female}(z)$$

In $\mathcal{FF}$:

$$\forall x_1\ \texttt{grf-wth-gdtrs}(x_1) \to \exists x_2\ \texttt{hasChld}(x_1, x_2) \wedge \exists x_3\ \texttt{hasChld}(x_2, x_3) \wedge \mathit{female}(x_3)$$

Note that the Forward Guarded Fragment $\mathcal{FGF} := \mathcal{GF} \cap \mathcal{FF}$ also captures $\mathcal{ALC}$.

The Forward Guarded Fragment $\mathcal{FGF}$ [B., JELIA 2021]: Our results

The Forward Guarded Fragment $\mathcal{FGF}$ [B., JELIA 2021]: Our results

- New, arguably elegant logic $\mathcal{FGF}$ over relational, equality-free signatures.

The Forward Guarded Fragment $\mathcal{FGF}$ [B., JELIA 2021]: Our results

- New, arguably elegant logic $\mathcal{FGF}$ over relational, equality-free signatures.
- $\mathcal{FGF}$ cannot express "bad guys": transitivity, self-loops, nominals and inverses.

The Forward Guarded Fragment $\mathcal{FGF}$ [B., JELIA 2021]: Our results

- New, arguably elegant logic $\mathcal{FGF}$ over relational, equality-free signatures.
- $\mathcal{FGF}$ cannot express "bad guys": transitivity, self-loops, nominals and inverses.

$$\varphi_{\mathrm{tr}(R)} = \forall x_1 \forall x_2 \forall x_3 \; R(x_1, x_2) \wedge R(x_2, x_3) \rightarrow R(x_1, x_3).$$

The Forward Guarded Fragment $\mathcal{FGF}$ [B., JELIA 2021]: Our results

- New, arguably elegant logic $\mathcal{FGF}$ over relational, equality-free signatures.
- $\mathcal{FGF}$ cannot express "bad guys": transitivity, self-loops, nominals and inverses.

$\varphi_{\mathrm{tr}(R)} = \forall x_1 \forall x_2 \forall x_3\ R(x_1, x_2) \land R(x_2, x_3) \to R(x_1, x_3)$.

$\varphi_{\mathrm{loop}(R)}(x_1) = R(x_1, x_1)$.

The Forward Guarded Fragment $\mathcal{FGF}$ [B., JELIA 2021]: Our results

- New, arguably elegant logic $\mathcal{FGF}$ over relational, equality-free signatures.
- $\mathcal{FGF}$ cannot express "bad guys": transitivity, self-loops, nominals and inverses.

$$\varphi_{\mathrm{tr}(R)} = \forall x_1 \forall x_2 \forall x_3 \ R(x_1, x_2) \wedge R(x_2, x_3) \rightarrow R(x_1, x_3).$$

$$\varphi_{\mathrm{loop}(R)}(x_1) = R(x_1, x_1).$$

$$\varphi_{\mathrm{inv}(S)=R} := \forall x_1 x_2 S(x_1, x_2) \leftrightarrow R(x_2, x_1)$$

The Forward Guarded Fragment $\mathcal{FGF}$ [B., JELIA 2021]: Our results

- New, arguably elegant logic $\mathcal{FGF}$ over relational, equality-free signatures.
- $\mathcal{FGF}$ cannot express “bad guys”: transitivity, self-loops, nominals and inverses.

$$\varphi_{\mathrm{tr}(R)} = \forall x_1 \forall x_2 \forall x_3 \; R(x_1, x_2) \land R(x_2, x_3) \to R(x_1, x_3).$$

$$\varphi_{\mathrm{loop}(R)}(x_1) = R(x_1, x_1).$$

$$\varphi_{\mathrm{inv}(S)=R} := \forall x_1 x_2 S(x_1, x_2) \leftrightarrow R(x_2, x_1)$$

$$\varphi_{\mathrm{unique}(A)} := \forall x_1 x_2 \; \underbrace{A(x_1) \land A(x_2)}_{\text{not guarded!}} \to x_1 = x_2$$

The Forward Guarded Fragment $\mathcal{FGF}$ [B., JELIA 2021]: Our results

- New, arguably elegant logic $\mathcal{FGF}$ over relational, equality-free signatures.
- $\mathcal{FGF}$ cannot express "bad guys": transitivity, self-loops, nominals and inverses.

$$\varphi_{\mathsf{tr}(R)} = \forall x_1 \forall x_2 \forall x_3\ R(x_1, x_2) \land R(x_2, x_3) \to R(x_1, x_3).$$

$$\varphi_{\mathsf{loop}(R)}(x_1) = R(x_1, x_1).$$

$$\varphi_{\mathsf{inv}(S)=R} := \forall x_1 x_2 S(x_1, x_2) \leftrightarrow R(x_2, x_1)$$

$$\varphi_{\mathsf{unique}(A)} := \forall x_1 x_2\ \underbrace{A(x_1) \land A(x_2)}_{\text{not guarded!}} \to x_1 = x_2$$

Theorem (B., JELIA 2021)

Knowledge-base SAT and CQ entailment for $\mathcal{FGF}$ is ExpTime-complete.

The Forward Guarded Fragment $\mathcal{FGF}$ [B., JELIA 2021]: Our results

- New, arguably elegant logic $\mathcal{FGF}$ over relational, equality-free signatures.
- $\mathcal{FGF}$ cannot express "bad guys": transitivity, self-loops, nominals and inverses.

$$\varphi_{\text{tr}(R)} = \forall x_1 \forall x_2 \forall x_3 \; R(x_1, x_2) \wedge R(x_2, x_3) \rightarrow R(x_1, x_3).$$

$$\varphi_{\text{loop}(R)}(x_1) = R(x_1, x_1).$$

$$\varphi_{\text{inv}(S)=R} := \forall x_1 x_2 S(x_1, x_2) \leftrightarrow R(x_2, x_1)$$

$$\varphi_{\text{unique}(A)} := \forall x_1 x_2 \; \underbrace{A(x_1) \wedge A(x_2)}_{\text{not guarded!}} \rightarrow x_1 = x_2$$

Theorem (B., JELIA 2021)

Knowledge-base SAT and CQ entailment for $\mathcal{FGF}$ is $\textsc{ExpTime}$-complete.

Harvesting from the results of Grädel and Bárány et al:

The Forward Guarded Fragment $\mathcal{FGF}$ [B., JELIA 2021]: Our results

- New, arguably elegant logic $\mathcal{FGF}$ over relational, equality-free signatures.
- $\mathcal{FGF}$ cannot express "bad guys": transitivity, self-loops, nominals and inverses.

$$\varphi_{\mathrm{tr}(R)} = \forall x_1 \forall x_2 \forall x_3\ R(x_1, x_2) \wedge R(x_2, x_3) \to R(x_1, x_3).$$

$$\varphi_{\mathrm{loop}(R)}(x_1) = R(x_1, x_1).$$

$$\varphi_{\mathrm{inv}(S)=R} := \forall x_1 x_2 S(x_1, x_2) \leftrightarrow R(x_2, x_1)$$

$$\varphi_{\mathrm{unique}(A)} := \forall x_1 x_2\ \underbrace{A(x_1) \wedge A(x_2)}_{\text{not guarded!}} \to x_1 = x_2$$

Theorem (B., JELIA 2021)

Knowledge-base SAT and CQ entailment for $\mathcal{FGF}$ is $\textsc{ExpTime}$-complete.

Harvesting from the results of Grädel and Bárány et al:

Corollary

Data complexity of KB SAT is NP-compl and coNP-compl for querying.

$\mathcal{FGF}$ has FMP and is finitely-controllable.

Two main ingredients: forward-types and HAFs

Two main ingredients: forward-types and HAFs

Definition (Forward type)

A (Σ, n)-*forward type* is a conjunction of atoms with n free-variables $\vec{x}_{1\ldots n}$, which for every relational symbol $\mathrm{R} \in \Sigma$ of arity $\ell = \mathrm{ar}(\mathrm{R}) \leq n$ and every index $1 \leq i \leq n+1-\ell$ contains either $\mathrm{R}(\vec{x}_{i\ldots i+\ell-1})$ or $\neg\mathrm{R}(\vec{x}_{i\ldots i+\ell-1})$.

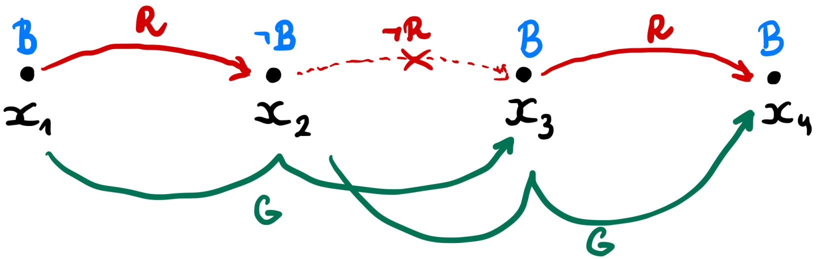
Two main ingredients: forward-types and HAFs

Definition (Forward type)

A (Σ, n)-*forward type* is a conjunction of atoms with n free-variables $\vec{x}_{1\dots n}$, which for every relational symbol $\mathrm{R} \in \Sigma$ of arity $\ell = \mathrm{ar}(\mathrm{R}) \leq n$ and every index $1 \leq i \leq n{+}1{-}\ell$ contains either $\mathrm{R}(\vec{x}_{i\dots i+\ell-1})$ or $\neg\mathrm{R}(\vec{x}_{i\dots i+\ell-1})$.

Blue B^1, Red R^2, Green G^3

$(\{R, G, B\}, 4)$-forward tp



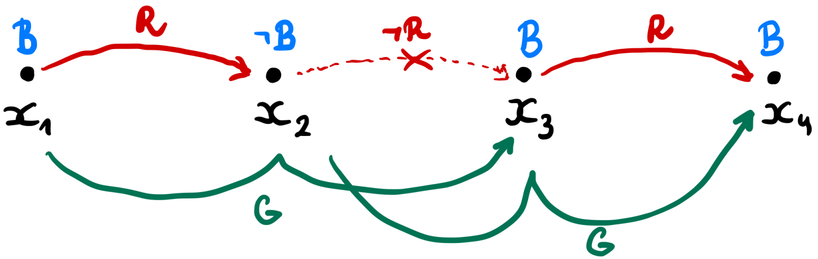
Two main ingredients: forward-types and HAFs

Definition (Forward type)

A (Σ, n)-*forward type* is a conjunction of atoms with n free-variables $\vec{x}_{1\ldots n}$, which for every relational symbol $\mathrm{R} \in \Sigma$ of arity $\ell = \mathrm{ar}(\mathrm{R}) \leq n$ and every index $1 \leq i \leq n+1-\ell$ contains either $\mathrm{R}(\vec{x}_{i\ldots i+\ell-1})$ or $\neg\mathrm{R}(\vec{x}_{i\ldots i+\ell-1})$.

Blue B^1, Red R^2, Green G^3

$(\{R, G, B\}, 4)$-forward tp



Lemma

The number of different (Σ, n)-types is $\leq 2^{|\Sigma| \cdot n^2}$.

The number of conjuncts in each (Σ, n)-type is $\leq |\Sigma| \cdot n$

Two main ingredients: forward-types and HAFs

Two main ingredients: forward-types and HAFs

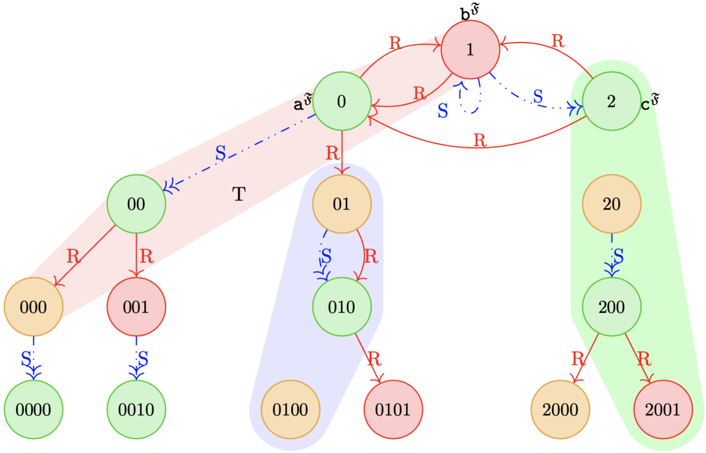
Definition (Higher-arity forests (HAFs))

There are forests in which (higher-arity) edges link roots in arbitrary way but other elements are connected in the level-by-level order.

Two main ingredients: forward-types and HAFs

Definition (Higher-arity forests (HAFs))

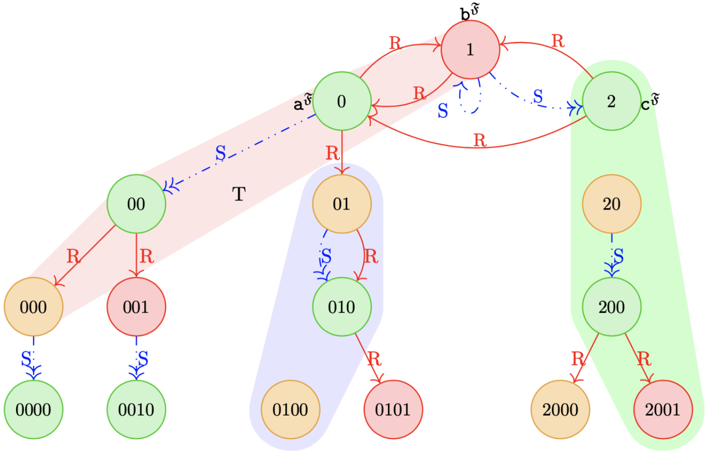
There are forests in which (higher-arity) edges link roots in arbitrary way but other elements are connected in the level-by-level order.



Two main ingredients: forward-types and HAFs

Definition (Higher-arity forests (HAFs))

There are forests in which (higher-arity) edges link roots in arbitrary way but other elements are connected in the level-by-level order.



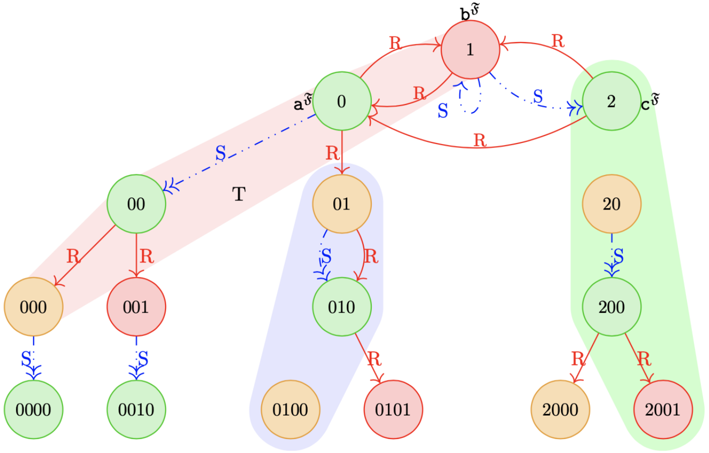
Lemma

Every satisfiable $\mathcal{FGF}$ knowledge base has a HAF (counter)model.

Two main ingredients: forward-types and HAFs

Definition (Higher-arity forests (HAFs))

There are forests in which (higher-arity) edges link roots in arbitrary way but other elements are connected in the level-by-level order.



Lemma

Every satisfiable $\mathcal{FGF}$ knowledge base has a HAF (counter)model.

Theorem (B., JELIA'21)

Knowledge-base SAT for $\mathcal{FGF}$ is $\textsc{ExpTime}$-complete.

Conclusions

Conclusions

Forward GF = formulae guarded but kept forward

Conclusions

Forward GF $=$ formulae guarded but kept forward

Theorem (B., JELIA 2021)

Knowledge-base SAT and CQ entailment for $\mathcal{FGF}$ is $\textsc{ExpTime}$-complete, also in the finite. Data complexity (co)NP-complete. FMP $+$ Fin-Control.

Conclusions

Forward GF = formulae guarded but kept forward

Theorem (B., JELIA 2021)

Knowledge-base SAT and CQ entailment for $\mathcal{FGF}$ is $\textsc{ExpTime}$-complete, also in the finite. Data complexity (co)NP-complete. FMP + Fin-Control.

Open problems and future research?

Conclusions

Forward GF = formulae guarded but kept forward

Theorem (B., JELIA 2021)

Knowledge-base SAT and CQ entailment for $\mathcal{FGF}$ is $\textsc{ExpTime}$-complete, also in the finite. Data complexity (co)NP-complete. FMP + Fin-Control.

Open problems and future research?

1. Understand model theory of Ordered/Fluted/Forward Fragment of $\mathcal{FO}$.

Conclusions

Forward GF $=$ formulae guarded but kept forward

Theorem (B., JELIA 2021)

Knowledge-base SAT and CQ entailment for $\mathcal{FGF}$ is $\textsc{ExpTime}$-complete, also in the finite. Data complexity (co)NP-complete. FMP $+$ Fin-Control.

Open problems and future research?

1. Understand model theory of Ordered/Fluted/Forward Fragment of $\mathcal{FO}$.

i.e. E-F Games, Craig Interpolation, Beth Definability, Preservation Theorems à la Łoś-Tarski

Conclusions

Forward GF = formulae guarded but kept forward

Theorem (B., JELIA 2021)

Knowledge-base SAT and CQ entailment for $\mathcal{FGF}$ is $\textsc{ExpTime}$-complete, also in the finite. Data complexity (co)NP-complete. FMP + Fin-Control.

Open problems and future research?

1. Understand model theory of Ordered/Fluted/Forward Fragment of $\mathcal{FO}$.

i.e. E-F Games, Craig Interpolation, Beth Definability, Preservation Theorems à la Łoś-Tarski

Ongoing work with Reijo Jaakkola, University of Tampere

Conclusions

Forward GF = formulae guarded but kept forward

Theorem (B., JELIA 2021)

Knowledge-base SAT and CQ entailment for $\mathcal{FGF}$ is $\textsc{ExpTime}$-complete, also in the finite. Data complexity (co)NP-complete. FMP + Fin-Control.

Open problems and future research?

1. Understand model theory of Ordered/Fluted/Forward Fragment of $\mathcal{FO}$.

i.e. E-F Games, Craig Interpolation, Beth Definability, Preservation Theorems à la Łoś-Tarski

Ongoing work with Reijo Jaakkola, University of Tampere

2. Study $\mathcal{FGF} + \mathcal{I}/\mathcal{O}/\mathcal{Q}$.

Conclusions

Forward GF = formulae guarded but kept forward

Theorem (B., JELIA 2021)

Knowledge-base SAT and CQ entailment for $\mathcal{FGF}$ is $\textsc{ExpTime}$-complete, also in the finite. Data complexity (co)NP-complete. FMP + Fin-Control.

Open problems and future research?

1. Understand model theory of Ordered/Fluted/Forward Fragment of $\mathcal{FO}$.

i.e. E-F Games, Craig Interpolation, Beth Definability, Preservation Theorems à la Łoś-Tarski

Ongoing work with Reijo Jaakkola, University of Tampere

2. Study $\mathcal{FGF} + \mathcal{I}/\mathcal{O}/\mathcal{Q}$.

3. Study $\mathcal{FGF}{+}\mu$ or $\mathcal{FGF}{+}\mathcal{S}$. Seem to behave nicer than $\mathcal{GF}{+}TG$

Conclusions

Forward GF $=$ formulae guarded but kept forward

Theorem (B., JELIA 2021)

Knowledge-base SAT and CQ entailment for $\mathcal{FGF}$ is $\textsc{ExpTime}$-complete, also in the finite. Data complexity (co)NP-complete. FMP $+$ Fin-Control.

Open problems and future research?

1. Understand model theory of Ordered/Fluted/Forward Fragment of $\mathcal{FO}$.

i.e. E-F Games, Craig Interpolation, Beth Definability, Preservation Theorems à la Łoś-Tarski

Ongoing work with Reijo Jaakkola, University of Tampere

2. Study $\mathcal{FGF} + \mathcal{I}/\mathcal{O}/\mathcal{Q}$.

3. Study $\mathcal{FGF}{+}\mu$ or $\mathcal{FGF}{+}\mathcal{S}$. Seem to behave nicer than $\mathcal{GF}{+}TG$

4. Effective algorithms, e.g. resolution-based/sequent proofs (with Tim Lyon).

Conclusions

Forward GF = formulae guarded but kept forward

Theorem (B., JELIA 2021)

Knowledge-base SAT and CQ entailment for $\mathcal{FGF}$ is $\textsc{ExpTime}$-complete, also in the finite. Data complexity (co)NP-complete. FMP + Fin-Control.

Open problems and future research?

1. Understand model theory of Ordered/Fluted/Forward Fragment of $\mathcal{FO}$.

i.e. E-F Games, Craig Interpolation, Beth Definability, Preservation Theorems à la Łoś-Tarski

Ongoing work with Reijo Jaakkola, University of Tampere

2. Study $\mathcal{FGF} + \mathcal{I}/\mathcal{O}/\mathcal{Q}$.

3. Study $\mathcal{FGF}{+}\mu$ or $\mathcal{FGF}{+}\mathcal{S}$. Seem to behave nicer than $\mathcal{GF}{+}TG$

4. Effective algorithms, e.g. resolution-based/sequent proofs (with Tim Lyon).

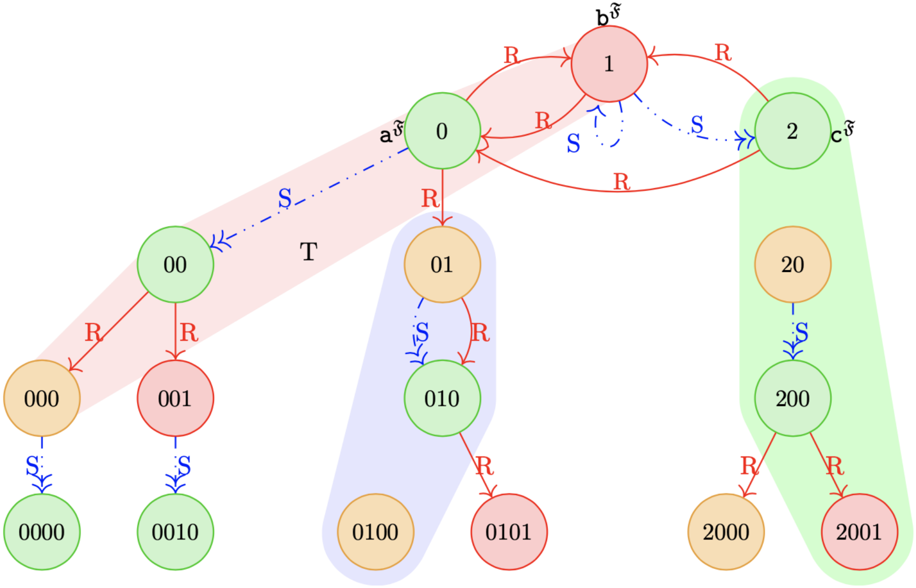
5. Forward TGDs (with Piotr Nalewaja).

Conclusions

Forward GF = formulae guarded but kept forward

Theorem (B., JELIA 2021)

Knowledge-base SAT and CQ entailment for $\mathcal{FGF}$ is $\textsc{ExpTime}$-complete, also in the finite. Data complexity (co)NP-complete. FMP + Fin-Control.



research?

ward Fragment of $\mathcal{FO}$.

ervation Theorems à la Łoś-Tarski

rsity of Tampere

icer than $\mathcal{GF}$+$\mathcal{TG}$

nt proofs (with Tim Lyon).

Thanks for attention!