Description Logics
Old results and new problems

Franz Baader
Theoretical Computer Science
RWTH Aachen
Germany

- Introduction to Description Logics (terminological KR languages, concept languages, KL-ONE-like KR languages, ...)
- Research in DL (historical overview)
- Connection with (simple) conceptual graphs
- New inference problems: unification and matching of concepts

Description logics

origin, ancestors

- Descend from structured inheritance networks [Brachman 78].
- Tried to overcome ambiguities in semantic networks and frames that were due to their lack of a formal semantics.
- Restriction to a small set of "epistemologically adequate" operators for defining concepts.
- Importance of well-defined basic inference procedures: subsumption and instance problem.
- First realization: system KL-ONE [Brachman&Schmolze 85], many successor systems (Classic, Crack, Fact, Flex, Kris, Loom, ...).
- First application: natural language processing; now also other domains (configuration of technical systems, databases, chemical engineering, medical terminology, ...)

Description Logic Systems

description language

- constructors for building complex concepts and roles out of atomic concepts and roles
- formal, logic-based semantics

TBox

defines terminology of the application domain

ABox

states facts about a specific "world"

knowledge base

reasoning component

- derive implicitly represented knowledge (e.g., subsumption)
- "practical" algorithms

Description language

examples of typical constructors:

$C \sqcap D$, $\neg C$, $\forall r . C$, $\exists r . C$, $(\geq n\ r)$

A man	Human $\sqcap$ $\neg$ Female $\sqcap$
that is married to a doctor, and	$\exists$ married-to . Doctor $\sqcap$
has at least 5 children,	($\geq$ 5 has-child) $\sqcap$
all of whom are professors.	$\forall$ has-child . Professor

TBox

definition of concepts

Happy-man = Human $\sqcap$ ...

ABox

properties of individuals

Happy-Man(John)

married-to(John,Mary)

Formal semantics

based on interpretations as in predicate logic

An interpretation I associates

- concepts C with sets C^I and
- roles r with binary relations r^I

such that the semantics of the constructors is respected; e.g.,

- $(C \sqcap D)^I = C^I \cap D^I$
- $(\geq n\ r)^I = \{ d \mid \#\{e \mid (d,e) \in r^I\} \geq n \}$
- $(\forall r . C)^I = \{ d \mid \forall e: (d,e) \in r^I \Rightarrow e \in C^I \}$
- ...

$I \models A = C$ iff $A^I = C^I$

$I \models C(a)$ iff $a^I \in C^I$

$I \models r(a,b)$ iff $(a^I, b^I) \in r^I$

Reasoning

makes implicitly represented knowledge explicit,
is provided as system service by the DL system, e.g.:

Satisfiability

Is a concept description C non-contradictory?

C is satisfiable iff there is an I such that $C^I \neq \emptyset$.

Subsumption

Is C a subconcept of D?

$C \sqsubseteq D$ iff $C^I \subseteq D^I$ for all interpretations I.

Instantiation

Is e an instance of C w.r.t. the given ABox $\mathcal{A}$?

$\mathcal{A} \models C(e)$ iff $e^I \in C^I$ for all models I of $\mathcal{A}$.

Focus of DL research

- decidability/complexity of reasoning
- requires restricted description languages
- systems and theoretical results available for various combinations of constructors

- application relevant concepts must be definable
- specific application domains may require specific language extensions
- new decidability/complexity results?

Reasoning feasible ⟷ versus ⟷ **Expressivity sufficient**

DL research

historical overview

- until 1985: mostly system development;
  expressive description languages, but no disjunction, negation, exist. quant.;
  use of so-called structural subsumption algorithms.

- 1985-1987: introduction of logic-based semantics;
  first complexity results (NP-hardness) by Levesque and Brachman;
  incompleteness of structural algorithms.

- 1988: Schmidt-Schauß and Smolka describe the first complete
  (tableau-based) subsumption algorithm for a non-trivial language;
  $\mathcal{ALC}$: propositionally closed (disjunction, negation, existential restrictions);
  subsumption as logical inference problem, reduced to satisfiability;
  complexity result: subsumption in $\mathcal{ALC}$ is PSPACE-complete.

DL research (continued)

- 1989: three papers [Patel-Schneider; Schild; Schmidt-Schauß] show undecidability of subsumption for description languages used in implemented DL systems.
- since 1989: development of tableau-based algorithms for a great variety of description languages (DFKI, Germany; University of Rome I, RWTH Aachen, ...); extended to the instance problem for ABoxes.
- 1989 - 1991: exact worst-case complexity of satisfiability and subsumption for various description languages (DFKI, Germany; University of Rome I).
- 1991: Schild notices a close connection between DLs and modal logics; $\mathcal{ALC}$ is just a syntactic variant of propositional multi-modal K; algorithms, complexity results from modal logics carry over.
- 1992-1995: development of very expressive Description Logics based on decidable extensions of K (University of Rome I); e.g., used to express semantic data models (ER, OO, ...).

DL research (continued)

- 1991-1998: close connection between DLs and decidable sub-classes of first-order logic:
  - $\mathcal{ALC}$ can be expressed within L2, i.e., first-order logic with two variables: decidable [Mortimer 75], NEXPTIME-complete [Grädel, V., K. 97]
  - number restrictions can be expressed in C2, i.e., the extension of L2 by counting quantifiers: decidable [Grädel, O., R. 97], in 2-NEXPTIME [Pacholski, S., T. 97]

- 1992-1998: optimization of DL systems based on complete (tableau-like) algorithms:
  - try to avoid explicit calls of subsumption algorithm during classification [Baader et. al 92, 94]; similar to techniques employed in CG systems [Ellis 91; Levinson 92].
  - optimization of subsumption algorithms [Giunchiglia & Sebastiani 96, 98; Horrocks 98; Patel-Schneider 98].

Connection with Conceptual Graphs

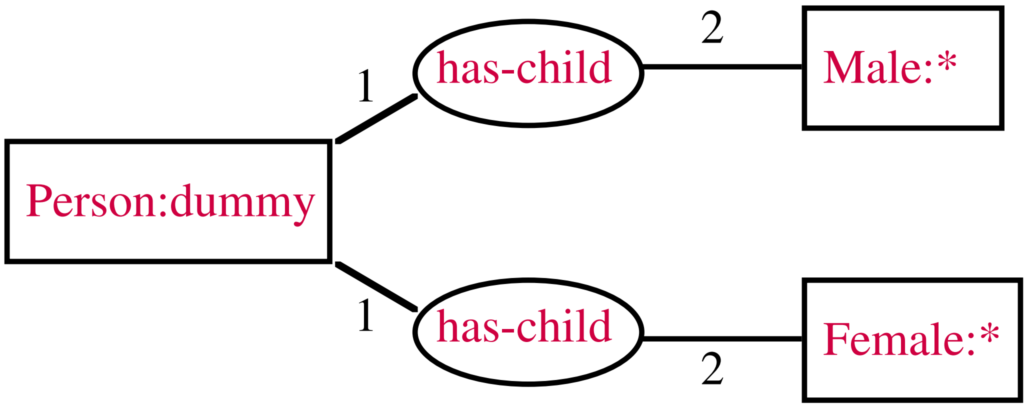
- Conceptual graphs have the "full power of first-order logic" [Sowa 84] . Thus, most of the description languages considered in DL can be expressed.
- What about reasoning? Does this connection provide us with graph-based reasoning methods for DL?
  - Possible way of finding and/or explaining incomplete (subsumption) algorithms?
  - Sub-class of CGs for which graph-based methods yield decision procedures, and thus complete algorithms?

Simple Conceptual Graphs

[Chein, Mugnier, Simonet KR'98]

can express concept descriptions built using conjunction ($\sqcap$) and existential restriction ($\exists\, r\,.\,C$).

Person $\sqcap$
$\exists$ has-child . Male $\sqcap$
$\exists$ has-child . Female



- Subsumption of descriptions corresponds to subsumption of SGs.
- Subsumption of SGs characterized by existence of projection.
- Testing for existence of projection is NP-complete.
- Subsumption of decriptions is polynomial since translation yields SGs that are trees.

New inference problems*

- Until recently, DL research concentrated on the traditional inference problems subsumption and instantiation.
- Building and maintaining larger knowledge bases requires support by new types of inference methods, e.g.:
  - unification of concepts: detect redundancies in KB
  - matching of concepts: prune large concept descriptions before printing them
- In the rest of the talk:
  - unification and matching in the simple DL $\mathcal{FL}_0$: conjunction $C \sqcap D$, value restriction $\forall r . C$
  - extension to larger languages

* Joint work with A. Borgida, R. Küsters, D. McGuinness, P. Narendran

Rheinisch-
Westfälische
Technische
Hochschule
Aachen
RWTH

Unification of concepts

Motivation

Situation very large terminology is built by several knowledge engineers over a long time period (our application: process engineering)

Testing for equivalence is not sufficient to find out whether two concept descriptions describe the same concept: different knowledge engineers

- introduce different concept names for the same (intuitive) concept:

  Masculine instead of Male

- model on different levels of granularity:

Man	as atomic concept name
$\mathsf{Human} \sqcap \mathsf{Male}$	as a concept term expressing the same concept
$\mathsf{Human} \sqcap \mathsf{Male} \sqcap \forall\,\mathsf{drinks}.\mathsf{Beer}$	Bavarian knowledge engineer

Unification of concepts

Definition

Set of concept names is partitioned into concept variables and concept constants:

- concept patterns may contain variables
- concept descriptions not

- substitution replaces concept variables by concept descriptions
- unifier of two concept patterns C and D: substitution σ such that

$$\boxed{\sigma(C) \equiv \sigma(D)}$$

i.e., $\sigma(C)^I = \sigma(D)^I$ for all interpretations I.

Example

Might the following concept descriptions denote the same concept?

$\forall \mathsf{child}.\forall \mathsf{child}.\mathsf{Rich} \sqcap \forall \mathsf{child}.\mathsf{RMR}$

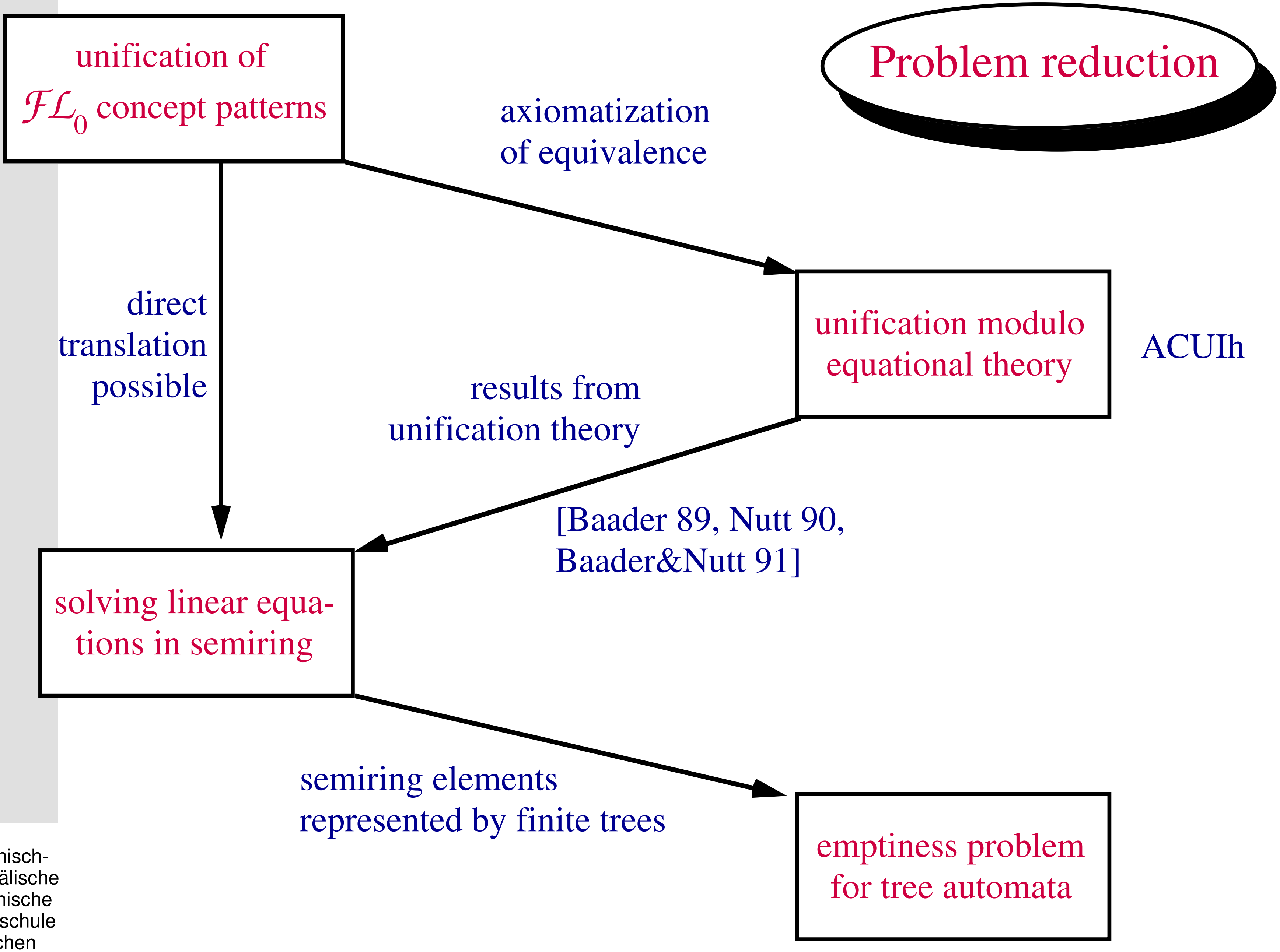
$\downarrow \quad \mathsf{RMR} \mapsto \qquad \sqcap \forall \mathsf{spouse}.\mathsf{Rich}$

$\forall \mathsf{child}.\forall \mathsf{child}.\mathsf{Rich} \sqcap$
$\forall \mathsf{child}.(\mathsf{Rich} \sqcap \forall \mathsf{spouse}.\mathsf{Rich})$

All grandchildren are rich and
all children are rich and married rich.

$\uparrow \quad \mathsf{ACR} \mapsto \forall \mathsf{child}.\mathsf{Rich}$

$\mathsf{ACR} \sqcap \forall \mathsf{child}.\mathsf{ACR} \sqcap \forall \mathsf{child}.\forall \mathsf{spouse}.\mathsf{Rich}$


Problem reduction
unification of
$\mathcal{FL}_0$ concept patterns
axiomatization
of equivalence
unification modulo
equational theory
ACUIh
direct
translation
possible
results from
unification theory
[Baader 89, Nutt 90,
Baader&Nutt 91]
solving linear equa-
tions in semiring
semiring elements
represented by finite trees
emptiness problem
for tree automata

The semiring corresponding to $\mathcal{FL}_0$

Elements:	finite sets of words over alphabet of role names e.g., Ø, {c, cs, ccs}, {s}, ...
Addition:	set union $\{c, cs, ccs\} \cup \{s\} = \{c, cs, ccs, s\}$
Multiplication:	element-wise concatenation $\{s\}\{c, cs, ccs\} = \{sc, scs, sccs\}$

Linear equations

S_i, T_i coefficients, X_i variables

$$S_0 \cup S_1X_1 \cup \ldots \cup S_nX_n = T_0 \cup T_1X_1 \cup \ldots \cup T_nX_n$$

unification of
$\mathcal{FL}_0$ concept patterns

direct translation:

solving linear equations in semiring

- Normal form for concept descriptions and patterns
- Characterization of equivalence of concept descriptions in normal form
- Translate this characterization into linear equations

Concept-centered NF in $\mathcal{FL}_0$

$\forall r.A \sqcap \forall r.(B \sqcap \forall r.A)$	$\forall r.(A \sqcap \forall r.A) \sqcap \forall r.B$
$\downarrow$ *distribute* $\forall r$ *over conjunction*	$\downarrow$
$\forall r.A \sqcap \forall r.B \sqcap \forall r.\forall r.A$	$\forall r.A \sqcap \forall r.\forall r.A \sqcap \forall r.B$
$\downarrow$ *collect value-restriction set for each concept*	$\downarrow$
$\forall \{r, rr\}.A \sqcap \forall \{r\}.B$	$\forall \{r, rr\}.A \sqcap \forall \{r\}.B$

- In $\mathcal{FL}_0$, equality of value-restriction sets characterizes equivalence.
- Value-restriction sets are finite sets of words over the alphabet of role names, i.e., elements of the semiring.

Translation of unification problem

into linear equations
over finite sets of words

$$C \equiv \forall K_1 . A_1 \sqcap \ldots \sqcap \forall K_n . A_n \sqcap \forall L_1 . X_1 \sqcap \ldots \sqcap \forall L_k . X_k$$

$$D \equiv \forall M_1 . A_1 \sqcap \ldots \sqcap \forall M_n . A_n \sqcap \forall N_1 . X_1 \sqcap \ldots \sqcap \forall N_k . X_k$$

Equation (A_i)

$$K_i \cup L_1 X_{1,i} \cup \ldots \cup L_k X_{k,i} = M_i \cup N_1 X_{1,i} \cup \ldots \cup N_k X_{k,i}$$

$X_{i,j}$ variables for finite sets of words

Theorem

The unification problem $C \equiv^? D$ is solvable iff

the formal language equations $(A_1), \ldots, (A_n)$ are each solvable.

unifier

$X \mapsto \quad \sqcap \forall s.R$

$Y \mapsto \forall c.R$

unification of concept patterns

$$\boxed{\forall c.\forall c.R \sqcap \forall c.X \equiv^{?} Y \sqcap \forall c.Y \sqcap \forall c.\forall s.R}$$

linear equation (R)

$$\boxed{\{cc\} \cup \{c\}X = \{cs\} \cup \{\varepsilon, c\}Y}$$

solution $X = \{\varepsilon, s\},\ Y = \{c\}$

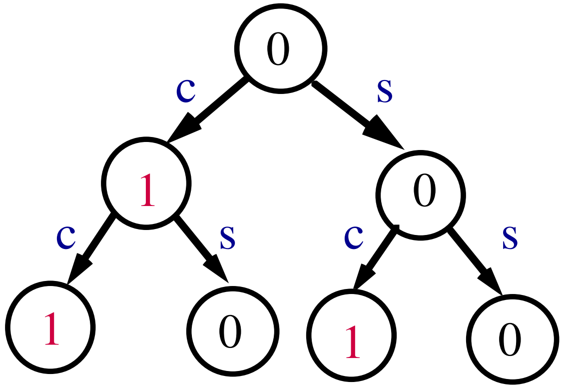
yields solution set $\{cc, c, cs\}$

Reduction to tree automata

solves mirrored equation

$$\{cc\} \cup X\{c\} \;=\; \{sc\} \cup Y\{\varepsilon, c\}$$

- Finite sets of words over an n-element alphabet can be represented by n-ary finite trees:



$\{cc, c, sc\}$ *mirrored solution*

- Top-down tree automaton tests for existence of solution set:
  - "guesses" the elements of the variables X_i
  - makes sure that the concatenation with the coefficients is realized

Theorem

[Baader&Narendran ECAI'98]

Unification of $\mathcal{FL}_0$-concept patterns is decidable.

Complexity

- Reduction to tree automata yields EXPTIME decision procedure:
  - size of tree automaton exponential in size of system of equations
  - emptiness problem for tree automata is polynomial
- Decision problem is EXPTIME-hard:
  - emptiness of intersection of m deterministic top down tree automata used for reduction

First results

for matching in DL

- Matching modulo subsumption [Borgida&McGuinness KR'96]
  - DL containing most of the CLASSIC constructs
  - polynomial matching algorithm
  - restriction on the syntactic form of patterns
- Matching modulo equivalence [Baader&Narendran ECAI'98]
  - as special case of unification in $\mathcal{FL}_0$
  - unlike unification, matching is polynomial for $\mathcal{FL}_0$
  - no restriction on the syntactic form of patterns

Extension of results

to larger description language allowing for
$\perp$, atomic negation, number restrictions

- Reduction of unification and matching problems to (extended) linear equations over finite sets of words still possible.
  - Main technical problem: appropriate treatment of inconsistency in the characterization of equivalence of concept descriptions.
- How to test the resulting linear equations for solvability?
  - Unification: open problem even for $\mathcal{FL}_0 + \perp$.
    The approach based on tree automata cannot work!
  - Matching: polynomial for $\mathcal{FL}_0 + \perp +$ atomic negation + number restrictions.
    Idea: compute largest "solution candidate" and test whether it is a solution.

Concept-centered NF in $\mathcal{FL}_0 + \bot$

$\forall r.\bot \sqcap \forall r.(B \sqcap \forall r.A)$ $\qquad$ $\forall r.(B \sqcap \forall r.B) \sqcap \forall r.\bot$

$\downarrow$ *distribute $\forall r$ over conjunction* $\downarrow$

$\forall r.\bot \sqcap \forall r.B \sqcap \forall r.\forall r.A$ $\qquad$ $\forall r.B \sqcap \forall r.\forall r.B \sqcap \forall r.\bot$

$\downarrow$ *collect value-restriction set for each concept* $\downarrow$

$\forall \{r\}.\bot \sqcap \forall \{rr\}.A \sqcap \forall \{r\}.B$ $\qquad$ $\forall \{r\}.\bot \sqcap \forall \emptyset.A \sqcap \forall \{r, rr\}.B$

In contrast to the situation for $\mathcal{FL}_0$, equality of value-restriction sets is no longer sufficient to characterize equivalence.

Concept-centered NF

characterization of equivalence in $\mathcal{FL}_0 + \bot$

$$C \equiv \forall L_0 . \bot \sqcap \forall L_1 . A_1 \sqcap \ldots \sqcap \forall L_n . A_n$$

$$D \equiv \forall M_0 . \bot \sqcap \forall M_1 . A_1 \sqcap \ldots \sqcap \forall M_n . A_n$$

L_i, M_i finite sets of words over the alphabet Σ of role names

Theorem

$C \equiv D$ iff $L_0\Sigma^* = M_0\Sigma^*$ and for $i = 1,\ldots, n$

$$L_i \cup L_0\Sigma^* = M_i \cup M_0\Sigma^*$$

Translation of matching problem

into linear equations
over finite sets of words

$$C \equiv \forall L_0 . \bot \sqcap \forall L_1 . A_1 \sqcap \ldots \sqcap \forall L_n . A_n$$

$$D \equiv \forall M_0 . \bot \sqcap \forall M_1 . A_1 \sqcap \ldots \sqcap \forall M_n . A_n \sqcap \forall N_1 . X_1 \sqcap \ldots \sqcap \forall N_k . X_k$$

Equation ($\bot$)

$$L_0\Sigma^* = M_0\Sigma^* \cup N_1X_{1,0}\Sigma^* \cup \ldots \cup N_kX_{k,0}\Sigma^*$$

Equation (A_i)

$$L_i \cup L_0\Sigma^* = M_i \cup N_1X_{1,i} \cup \ldots \cup N_kX_{k,i} \cup L_0\Sigma^*$$

$X_{i,j}$ variables for
finite sets of words

Theorem

The matching problem $C \equiv^? D$ is solvable iff

the formal language equations $(\perp)$, (A_1), ..., (A_n) are each solvable.

How to test solvability of $(\perp)$, (A_1), ..., (A_n) ?

- Compute largest "solution candidate".
- Test whether this candidate is indeed a solution.
- Both steps only require "easy" computations on finite sets of words (polynomial).

Conclusion

- Standard inference problems (subsumption, instantiation) well-investigated.
  - Decidability and complexity results for a great variety of description languages, including very expressive ones.
  - Efficient implementations of decision procedures available.
- Research on non-standard inference problems (unification, matching, ...) is just beginning:
  - Unification: decidability result only for the small language $\mathcal{FL}_0$; high complexity; unification in larger languages might be easier!?
  - Matching: polynomial for language that is expressive enough for applications.
  - Other applications for matching and/or unification, e.g., integration of heterogeneous databases?