Engineering Logical Algorithms
Using Saturation-Based Theorem Proving

Yevgeny L. Kazakov
ykazakov@mpi-inf.mpg.de

Automated theorem provers (ATP) are typically used as “black boxes”. They give no guarantee that a given problem will be solved within a particular amount of time.

On the other hand, the techniques underlying such theorem provers can be used to design special procedures for particular classes of problems, which can give such guarantees.

Description Logics (DLs) are families of languages used for knowledge representation and reasoning. Reasoning in description logics amounts to evaluation of queries over a database consisting of terminological definitions (**TBox**) and assertions about individuals (**ABox**).

TBox	**ABox**	**Query**
$A \doteq C$	Ind : C $(\mathsf{In}_1, \mathsf{In}_2)$: R	?− $C_1 \sqsubseteq C_2$?− Ind : C
Man $\doteq$ Human $\sqcap$ Male Parent $\doteq$ Human $\sqcap$ $\exists$has-child.Human Father $\doteq$ Man $\sqcap$ $\exists$has-child.Human Grandfather $\doteq$ Man $\sqcap$ $\exists$has-child.Parent	*John* : Man *Bill* : Father (*John*, *Bill*) : has-child	?− Grandfather $\sqsubseteq$ Father ?− *John* : Grandfather

For reasoning in DLs using ATPs, one needs to translate the input problem to clauses.

Such translation can be provided via the standard semantics for DL-constructors:

Clause Types

$\neg A(x) \vee B(x)$ $\neg A(x) \vee \neg B(x) \vee C(x)$ $\neg A(x) \vee B(f(x))$ …	D1(A,B) D2(A,B,C) C1(A,B,f)	$C(\mathsf{Ind})$ $R(\mathsf{In}_1,\mathsf{In}_2)$	C3(C,Ind) C4$(R,\mathsf{In}_1,\mathsf{In}_2)$	$C_1(\mathsf{q})$ $\neg C_2(\mathsf{q})$ $\neg C(\mathsf{Ind})$	C3(C_1,q) D4(C_2,q) D4(C_2,Ind)
$\neg$Man(x) $\vee$ Human(x) … $\neg$Human(x) $\vee$ $\neg$Male(x) $\vee$ Man(x) $\neg$Father(x) $\vee$ Human(child(x)) …		Man(*John*) Father(*Bill*) has-child(*John*, *Bill*)		Grandfather(P) $\neg$Father(P) $\neg$Grandfather(*John*)	

Possible Inferences Between Clauses

C1(A,B,f), D1(B,C) $\vdash$ C1(A,C,f) C1(A,B,f), D2(B,C,D) $\vdash$ D3(A,C,f) …	C3(A,Ind), D1(A,B) $\vdash$ C3(B,Ind)	C3(A,q), D4(A,q) $\vdash \bot$ …
$\neg A(x) \vee \underline{B(f(x))}$, $\underline{\neg B(x)} \vee C(x)$ $\vdash \neg A(x) \vee C(f(x))$ …	$\underline{A(\mathsf{Ind})}$, $\underline{\neg A(x)} \vee B(x)$ $\vdash B(\mathsf{Ind})$	$\underline{A(\mathsf{q})}$, $\underline{\neg A(\mathsf{q})} \vdash \square$

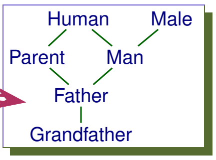
Datalog Program

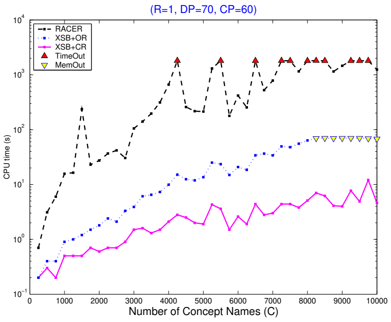
T1. C1(A,C,f) ← C1(A,B,f), D1(B,C).
T2. D3(A,C,f) ← C1(A,B,f), D2(B,C,D).
……

A1. C3(B,Ind) ← C3(A,Ind), D1(A,B).
……

Q1. **Subsumes**(B,A) ← D1(A,B).
Q2. **Instance**(Ind,B) ← C3(A,Ind).
……

By encoding inferences as datalog rules, one gets a considerable speed-up over conventional tableau-based procedures for DLs.





MAX-PLANCK-GESELLSCHAFT

max planck institut
informatik