FaCT and iFaCT

Ian Horrocks
University of Manchester, Manchester, UK
horrocks@cs.man.ac.uk

FaCT (**Fa**st **C**lassification of **T**erminologies) is a Description Logic (DL) classifier which has been implemented as a test-bed for a sound and complete tableaux satisfiability/subsumption testing algorithm. FaCT's novelty lies in its relatively expressive logic and its highly optimised implementation of the tableaux algorithm. iFaCT is an extension of FaCT that supports reasoning with inverse roles. The resulting logic is particularly interesting as it no longer has the finite model property.

Language

The logics implemented in FaCT and iFaCT are both based on $\mathcal{ALC}_{R^+}$, an extension of $\mathcal{ALC}$ to include transitive roles [Sattler, 1996]. For compactness, this logic will be called $\mathcal{S}$ (due to its relationship with the proposition multi-modal logic $\mathbf{S4_{(m)}}$ [Schild, 1991]). FaCT extends $\mathcal{S}$ with a hierarchy of roles and functional roles (attributes) to give $\mathcal{SHF}$, while iFaCT adds inverse roles to give $\mathcal{SHIF}$.

The constructs used by the two logics are described in Figure 1, where A is an atomic concept, R and S are roles, C and D are concepts, $\mathbf{R}_+$ is the set of transitive role names and $\mathbf{F}$ is the set of functional role names (there is an additional restriction that $\mathbf{R}_+$ and $\mathbf{F}$ must be disjoint). The meaning of concepts and roles is given by an *interpretation* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, consisting of a set $\Delta^{\mathcal{I}}$, called the *domain* of $\mathcal{I}$, and a function $\cdot^{\mathcal{I}}$ which maps every concept to a subset of $\Delta^{\mathcal{I}}$ and every role to a subset of $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$ such that the properties in Figure 1 are satisfied.

It is easy to show that $\mathcal{SHIF}$ does not have the finite model property: given $R \in \mathbf{R}_+$ (R is transitive), $F \in \mathbf{F}$ (F is functional) and $F \sqsubseteq R$, then the concept

$$\neg C \sqcap \exists F^-.C \sqcap \forall R^-.(\exists F^-.C)$$

is satisfiable, but all its models must contain in infinite sequence of individuals, each related to a single successors by an F^- role, and each satisfying $C \sqcap \exists F^-.C$, the $\exists F^-.C$ term being propagated along the sequence by the transitive super-role R. Terminating the sequence in a cycle back to the first element would result in a contradiction between C and $\neg C$, while a cycle back to any subsequent element would conflict with the functionality of F.

Both FaCT and iFaCT accept a variety of different input syntax, including a format compatible with the Kris system [Baader and Hollunder, 1991], and provide a wide range of functions and macros for constructing, classifying and querying a knowledge bases (KBs). In both systems, KBs can contain arbitrary (general concept inclusion) axioms, as well as the usual concept and role introduction axioms.

Other Features

Due to correspondences with propositional (multi) modal logics, FaCT can also be used as a reasoner for $\mathbf{K_{(m)}}$, $\mathbf{KT_{(m)}}$, $\mathbf{K4_{(m)}}$ and $\mathbf{S4_{(m)}}$; an interface is provided for testing the satisfiability of formulae in any of these logics.

In addition, iFaCT has an optional interface that allows it to reason (via a satisfiability maintaining mapping) with concepts and relationships (of arbitrary arity) in DLR, a logic which can be used to reasoning about database schemata [Calvanese *et al.*, 1998a].

Implementation

FaCT and iFaCT are implemented in Common Lisp, and have run successfully with several commercial and free lisps, including Allegro, Liquid (formerly Lucid), Lispworks and GNU.

Both FaCT and iFaCT transform subsumption problems into satisfiability problems, and solve these using sound and complete tableaux algorithms. Both algorithms use a form of loop checking called *blocking* to guarantee termination. In iFaCT, a special *pair-wise dynamic* blocking technique is used to deal with the additional complexities of inverse roles [Horrocks and Sattler, to appear]. Using this technique, the algorithm is able to generate finite representations of (possibly) infinite tableaux.

Construct Name	Syntax	Semantics	
atomic concept	A	$A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$	
atomic role	R	$R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$	
transitive role	$R \in \mathbf{R}_+$	$R^{\mathcal{I}} = (R^{\mathcal{I}})^+$	
conjunction	$C \sqcap D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$	$\mathcal{S}$
disjunction	$C \sqcup D$	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$	
negation	$\neg C$	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$	
exists restriction	$\exists R.C$	$\{x \mid \exists y.\langle x, y\rangle \in R^{\mathcal{I}} \text{ and } y \in C^{\mathcal{I}}\}$	
value restriction	$\forall R.C$	$\{x \mid \forall y.\langle x, y\rangle \in R^{\mathcal{I}} \text{ implies } y \in C^{\mathcal{I}}\}$	
role hierarchy	$R \sqsubseteq S$	$R^{\mathcal{I}} \subseteq S^{\mathcal{I}}$	$\mathcal{H}$
inverse role	R^-	$\{\langle x, y\rangle \mid \langle y, x\rangle \in R^{\mathcal{I}}\}$	$\mathcal{I}$
functional role	$R \in \mathbf{F}$	$\{\langle x, y\rangle, \langle x, z\rangle\} \subseteq R^{\mathcal{I}} \text{ implies } y = z$	$\mathcal{F}$

Figure 1: Syntax and semantics of the $\mathcal{SHF}$ and $\mathcal{SHIF}$

In order to make the FaCT system usable in realistic DL applications, a wide range of optimisation techniques are used in the implementation of the $\mathcal{SHF}$ satisfiability testing algorithm. These include axiom absorption, lexical normalisation, semantic branching search, simplification, dependency directed backtracking, heuristic guided search and caching. The use of these (and other) optimisation techniques has now become standard in tableaux implementations [Patel-Schneider, 1998, Haarslev *et al.*, 1998].

Performance

FaCT has performed well in a number of tests, including those using randomly generated or hand crafted formulae in some propositional modal logic. However, FaCT performs especially well when classifying KBs as several of its optimisations are specifically designed to take advantage of their typical structure.

Initial experiments with iFaCT have been encouraging. These experiments used a KB representing (fragments of) database schemata and inter schema assertions from a data warehousing application [Calvanese *et al.*, 1998a] (a slightly simplified version of the proposed encoding was used to generate a $\mathcal{SHIF}$ KB). iFaCT was able to classify this KB, which contains 19 concepts and 42 axioms, in less than 0.1s of (266MHz Pentium) CPU time. In contrast, eliminating inverse roles using an embedding technique [Calvanese *et al.*, 1998b] gives an equisatisfiable FaCT KB with an additional 84 axioms, but one which FaCT is unable to classify in 12 hours of CPU time. This is because the embedding generates general inclusion axioms (of the form $C \sqsubseteq D$, where C and D are arbitrary concepts) of a kind that are not, in general, amenable to absorption [Horrocks, 1998], an optimisation which is crucial for efficient reasoning w.r.t. general KBs.

The current implementation of iFaCT is a relatively naive modification of the FaCT system: it does not use the more efficient form of pair-wise blocking described in [Horrocks *et al.*, 1998], it does not include any special optimisations to deal with inverse roles (or take advantage of their absence), and some optimisations that would require modification in the presence of inverse roles are instead simply disabled. As a result, the performance of iFaCT is significantly worse than that of FaCT w.r.t. FaCT KBs (those not containing inverse roles).

Availability

FaCT and iFaCT are available (under the GNU general public license) via the WWW at http://www.cs.man.ac.uk/ horrocks.

Future Plans

Work is underway to add a (highly optimised) ABox reasoner to the FaCT system [Tessaris and Gough, to appear], and to provide a CORBA based client-server interface. Apart from this, further development of FaCT and iFaCT is not planned. Instead, a cooperative effort is underway to develop a new system that will support core reasoners for a range of expressive DLs. Amongst these will certainly be $\mathcal{SHIQ}$, a DL that augments $\mathcal{SHIF}$ with a qualifying number restrictions [Hollunder and Baader, 1991]. This logic will allow the complete encoding (via DLR) of and reasoning with database schemata (in particular, extended entity relationship models).

References

[Baader and Hollunder, 1991] F. Baader and B. Hollunder. KRIS: Knowledge representation and inference system. *SIGART Bulletin*, 2(3):8–14, 1991.

[Calvanese *et al.*, 1998a] D. Calvanese, G. De Giacomo, M. Lenzerini, D. Nardi, and R. Rosati. Source integration in data warehousing. In *Proc. of DEXA-98*, pages 192–197, 1998.

[Calvanese *et al.*, 1998b] D. Calvanese, G. De Giacomo, and R. Rosati. A note on encoding inverse roles and functional restrictions in $\mathcal{ALC}$ knowledge bases. In *Proc. of DL-98*, pages 69–71, 1998.

[Haarslev *et al.*, 1998] V. Haarslev, R. Möller, and A.-Y. Turhan. Implementing an $\mathcal{ALCRP}(\mathcal{D})$ abox reasoner – progress report. In *Proc. of DL-98*, pages 82–86, 1998.

[Hollunder and Baader, 1991] B. Hollunder and F. Baader. Qualifying number restrictions in concept languages. In *Proc. of KR'91*, pages 335–346, 1991.

[Horrocks and Sattler, to appear] I. Horrocks and U. Sattler. A description logic with transitive and inverse roles and role hierarchies. *Journal of Logic and Computation*, to appear.

[Horrocks *et al.*, 1998] I. Horrocks, U. Sattler, and S. Tobies. A PSPACE-algorithm for deciding $\mathcal{ALCI}_{R^+}$-satisfiability. Technical Report 98-08, LuFg Theoretical Computer Science, RWTH Aachen, 1998.

[Horrocks, 1998] I. Horrocks. Using an expressive description logic: FaCT or fiction? In *Proc. of KR'98*, pages 636–647, 1998.

[Patel-Schneider, 1998] P. F. Patel-Schneider. DLP system description. In *Proc. of DL-98*, pages 87–89, 1998.

[Sattler, 1996] U. Sattler. A concept language extended with different kinds of transitive roles. In *20. Deutsche Jahrestagung für Künstliche Intelligenz*, number 1137 in LNAI, pages 333–345, 1996.

[Schild, 1991] K. Schild. A correspondence theory for terminological logics: Preliminary report. In *Proc. of IJCAI-91*, pages 466–471, 1991.

[Tessaris and Gough, to appear] S. Tessaris and G. Gough. Abox reasoning with transitive roles and axioms. In *Proc. of DL'99*, to appear.