Automated Reasoning for Artificial Intelligence
Introduction to Description Logic

Final assignment

Deadline: June 30, 2011
Presentations: June 28, 2011

A complete assignment must include the following components:

1. a LoTREC *.xml file containing all required implementations, as specified in the particular tasks [LoTREC menu: `Logic/`texttttSave as...],
2. a report in a *.pdf file with all required solutions, as specified in the particular tasks,
3. the final presentation.

The solutions to the 4 tasks described below will contribute to the overall grade in the proportions 1) 40% 2) 20% 3) 20% 4) 20%.

1 Tableau algorithm

Use the LoTREC toolkit for implementing a tableau algorithm for deciding **concept satisfiability w.r.t. TBox** in the DL $\mathcal{ALC}$. Recall, that an instance of this problem is defined as:

Definition 1 *Given a pair $(\mathcal{T}, C)$, where $\mathcal{T}$ is a TBox and C a concept in $\mathcal{ALC}$, decide whether there exists a model $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ of $\mathcal{T}$ such that $C^{\mathcal{I}} \neq \emptyset$.*

(†) For simplicity, you can take the following assumptions:

- every TBox axiom in $\mathcal{T}$ is of the form $\top \equiv D$, where D is a concept in *Negation Normal Form*,
- the concept C is in *Negation Normal Form*.

To accomplish the task you will need to address the following points:

1. declare all constructors of $\mathcal{ALC}$ + auxiliary constructors for handling the input*) [LoTREC tab: `Connectors`].
2. define the rules of the tableau algorithm for $\mathcal{ALC}$ + auxiliary rules for handling the input*) [LoTREC tab: `Rules`].

3. define the strategy of using your rules during the run of the algorithm [LoTREC tab: `Strategies`].

*) Note, that LoTREC accepts only a single formula on the input, therefore you first need to choose your own syntax for "encoding" an instance of the problem as a single formula and further specify suitable rules for interpreting this encoding.

The tableau should terminate on every well-formed input. All individuals (LoTREC `nodes`) that are blocked by means of the blocking rule should be given an explicit label "BLOCK" on the branch (LoTREC `pre-model`) and an edge (LoTREC `link`) with label "Blocks" from the individual which warrants the blocking condition. Such individuals should not obtain successors. All individuals that contain a clash should be given an explicit label "CLASH", while the branch containing it should be prevented from further developing. The resulting tableau tree should contain either an open branch (i.e. saturated branch with no individuals containing the label CLASH) whenever the answer to the problem is positive (C *is satisfiable w.r.t.* $\mathcal{T}$) or otherwise, if the answer is negative (C *is not satisfiable w.r.t.* $\mathcal{T}$) — closed branches only (i.e. branches with an individual containing the label CLASH).

Output:

1. Provide an *.xml file containing your implementation of tableau algorithm
2. In your report, define the mapping from the DL syntax to the syntax used in LoTREC, plus explanation for auxiliary connectives, like the one presented in the example below (note that the choice of the syntax for your representation is totally up to you. We just need to be able to understand how a concept satisfiability problem in the DL $\mathcal{ALC}$ is represented in your implementation).

Example: We define a *sample* mapping from the DL syntax to the syntax used in LoTREC as follows:

1)	$\neg C$	$C \sqcap D$	$C \sqcup D$	$\exists r.C$	$\forall r.C$	$\top \equiv D$
2)	`not` C	`and` C D	`or` C D	`some` r C	`only` r C	`tbox` D
3)	not C	C and D	C or D	r some C	r only C	$\top = D$

where: 1) $\mathcal{ALC}$ syntax; 2) LoTREC connective; 3) LoTREC display

We write BOT instead of the bottom symbol $\bot$ and TOP instead of $\top$. Further we define two auxiliary connectives:

1. representation of a list of formulas φ, ψ, resp. $\varphi_1, \ldots, \varphi_n$
   - LoTREC connective: `add` φ ψ, resp. `add add` $\ldots \varphi, \ldots, \varphi_n$.
   - LoTREC display: φ & ψ, resp. φ_1 & $\ldots$ & φ_n.
2. representation of an instance of the problem:

- LoTREC connective: `input` $\mathcal{T}$ C
- LoTREC display: INPUT: TBox = $\mathcal{T}$; Concept = C

Based on these mappings, Figure 1 and 2 illustrate a possible display of a closed and an open branches of a tableau tree in LoTREC for the following problem:

Problem instance: $(\{\top \equiv \exists s.C, \top \equiv \forall r.(\neg C \sqcup D)\}, \exists r.(C \sqcap D))$
LoTREC input: `input add tbox some S C tbox only R or not C D some R and C D`

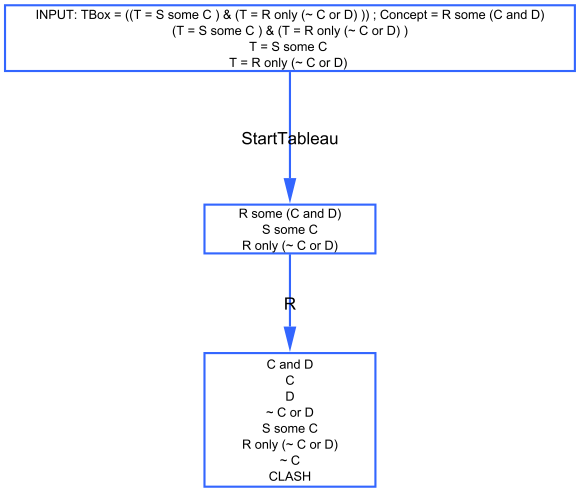

Figure 1: A closed branch of a tableau generated in LoTREC.

2 Reasoning problems

Solve the following decision problems supporting yourself with your implementation of tableau algorithm.

1. Is $\exists r.D$ satisfiable w.r.t. $\mathcal{T} = \{\top \equiv \exists s.C, \top \equiv \forall r.(\bot \sqcup E)\}$?
2. Is $D \sqcap E$ subsumed by $\exists r.B$ in $\mathcal{T} = \{C \sqsubseteq \neg A,\ D \sqsubseteq \forall r.(A \sqcup B),\ E \sqsubseteq \exists r.C\}$?
3. Is the ABox $\{C(a)\}$ consistent w.r.t. $\mathcal{T} = \{\top \equiv \forall r.B \sqcap \forall s.C,\ \top \equiv \neg\forall r.(\neg C \sqcap B),\ \top \equiv \exists s.\top\}$?

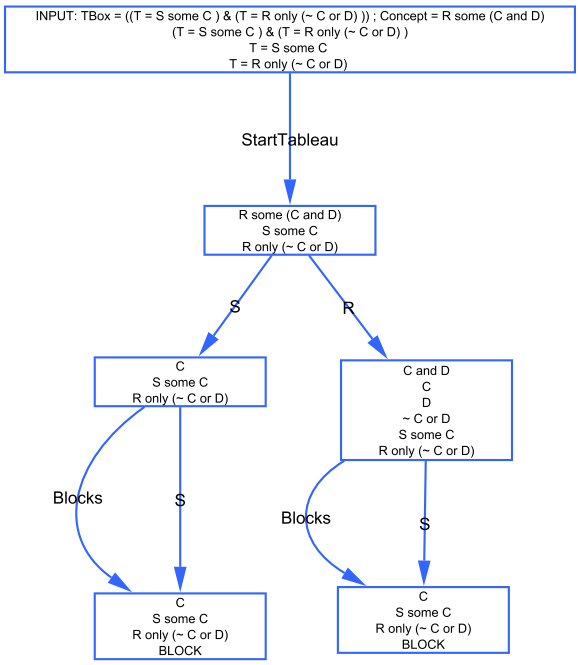

Figure 2: An open branch of a tableau generated in LoTREC.

Output: Your solution to every problem above should be reported in the following template:

1. State the problem.
2. Reduce the problem to the corresponding concept satisfiability problem (as shown during the lecture).
3. Apply the necessary syntactic transformations to the TBox and the concept so that the conditions (†), listed in the previous task, are satisfied.
4. Translate the result into the input formula for your tableau implementation. Please, type and save this formula also in your implementation file *.xml [LoTREC tab: `Predefined Formulas`]
5. Use your implementation of the tableau to compute a tableau tree for this formula.
6. State the result of the computation (is the tableau closed or open?) If the tableau is open include a picture of one of its open branches in the report [LoTREC menu: `Premodels/Export Premodel...`].

7. Based on the result of the computation provide the answers to the following problems: 1) the original decision problem; 2) the corresponding concept satisfiability problem that you solved with the tableau algorithm.

3 Strategies

Please elaborate on the following two questions.

Question 1

During the lecture we mentioned that in order to guarantee termination and completeness of the algorithm the blocking rule must be given a specific position in the ordering of the tableau rules. Namely, it should be used only when no other rules, except for possibly the existential rule, are applicable on the branch.

Output: Provide an intuitive explanation of why completeness might be lost in case this strategy is violated. To support your argument do the following:

- give an example of an alternative strategy [LoTREC tab: `Strategies`], which damages completeness. Name this strategy "WrongBlocking" and save it in your implementation file *.xml.
- define a simple concept satisfiability problem and feed it to your tableau implementation to illustrate the effect of this strategy. Save the input of this problem in the implementation file as well.
- discuss the loss of completeness in this case, i.e. show that although your concept is unsatisfiable, the tableau returns the opposite answer.

Hint: you can include and discuss a picture of one branch of the tableau, which is left open due to blocking, but which otherwise would have to be closed if blocking was not applied prematurely.

Question 2

Except for the requirement stated in the previous question, the ordering of rules can be in principle arbitrary, without affecting soundness, completeness or termination of the algorithm, provided they are used in a fair manner (i.e. if some rule is applicable then it will be eventually applied). However, some orderings seem to result in on average more space-efficient (and thus also time-efficient) algorithms.

Output: Propose an ordering in your report, which you believe should on average generate smaller tableau trees and give an intuitive justification for your proposal. Save the ordering as a strategy in your implementation file *.xml under name "EfficientOrdering". To support your argument, discuss your observations on a concrete example. Specify a suitable input problem and compare the

tableau trees generated on it by your efficient ordering and by alternative, non-efficient variants. Indicate the criteria you use for assessing the space-efficiency of the algorithms.

4 Extensions

Below we shortly describe three expressive features of Description Logics which go beyond the expressiveness of $\mathcal{ALC}$ and were not covered in depth during the lecture. Choose two out of them and propose a way of extending your implementation of the tableau algorithm, so that the algorithm remains sound, complete and terminating with respect to the extended logics (i.e. $\mathcal{ALC}$ extended with the selected features).

Output:

1. In your report, define the mapping from the additional DL syntax to the syntax used in LoTREC (similarly as the first task of this assignment). Describe and give the rationale behind the additional rules that you included in order to handle the new constructs.

2. Extend your *.xml implementation file accordingly. Save the extended strategy under the name corresponding to the names of the chosen extensions (e.g. `ALCwithInversesTransitiveRoles`).

Role inverses

Role inverses are used to express relations which are the exact opposites of the given roles. For an arbitrary role r its inverse is denoted by r^-. For instance, the concept $\mathit{Student} \sqcap \exists \mathit{like}.(\forall \mathit{like}^-.\neg \mathit{Student})$ describes the set of all these individuals which are students and *like* something which *is liked* only by non-students (obviously this concept is unsatisfiable). The semantics of a role inverse, under an interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, is defined as $(r^-)^{\mathcal{I}} = \{(x, y) \in \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \mid (x, y) \in r^{\mathcal{I}}\}$.

For simplicity, in your implementation you might assume that whenever r^- occurs in a DL formula, then r is a role name (i.e. we exclude the possibility of using nested inverses).

The extension of $\mathcal{ALC}$ with inverse roles is known as $\mathcal{ALCI}$.

Transitive roles

Some role names might be declared as transitive roles. The set of such roles is assumed to be disjoint from the regular role names and known in advance. For instance, we might want to consider the role *hasPart* as a transitive role, and thus observe that the concept $\exists \mathit{hasPart}.(\exists \mathit{hasPart}.(\exists \mathit{hasPart}.\mathit{Component}))$ is subsumed by $\exists \mathit{hasPart}.(\exists \mathit{hasPart}.\mathit{Component})$ and further by $\exists \mathit{hasPart}.\mathit{Component}$. The semantics of a transitive role r, under an interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, must

simply satisfy the additional condition: $\forall x, y, z : (x, y) \in r^{\mathcal{I}} \wedge (y, z) \in r^{\mathcal{I}} \rightarrow (x, z) \in r^{\mathcal{I}}$.

In your implementation you might distinguish the transitive roles from the normal roles by using some operator as a constant prefix for the former type of roles, e.g. + as in $+r$, denoting that $+r$ should be consistently interpreted as a transitive role (note, that + is not a role constructor).

The extension of $\mathcal{ALC}$ with transitive roles is known as $\mathcal{S}$.

Role hierarchies

Next to concept inclusions and equivalences, TBoxes might also include simple role inclusions of the form $r \sqsubseteq s$ which state that the role r is a "subrole" of the role s. For instance, *painted* $\sqsubseteq$ *created* asserts that whenever some individual x painted some y then x created y. Given this axiom, we can immediately infer that the concept $\exists \mathit{painted}.\mathit{Painting} \sqcap \forall \mathit{created}.\neg \mathit{Painting}$ is unsatisfiable. Formally, a role inclusion $r \sqsubseteq s$ is satisfied in an interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ *iff* $r^{\mathcal{I}} \subseteq s^{\mathcal{I}}$. An interpretation is a model of a role inclusion if the role inclusion is satisfied in that interpretation. Consequently, a model of a TBox is an interpretation which satisfies all its axioms, including all the role inclusions.

The extension of $\mathcal{ALC}$ with role hierarchies is known as $\mathcal{ALCH}$.