Henning Christiansen
24 September 2007 – small correction 25 Sep 2007

An exercise with Constraint Handling Rules

We consider a constraint solver to be used in a system that produces knowledge bases from text in natural language. The example is taken from two papers [1, 2] which consider generation of UML diagrams from use case text; however, here we ignore the grammatical part and generation of diagrams, and consider only the constraint solver that governs the knowledge base. An implemented system is reported, implemented Prolog and CHR.

Example 1:
From "A dog has four legs and one tail", the system generates to following constraints.

class(dog), class(leg), class(tail), property(dog,leg:4), property(dog,tail:1)

They are produced by grammar rules written in Prolog which "cast off" such constraints. What is interesting for us is what happens when there are different constraints about related entities, and this is where a constraint solver is needed.

Example 2:
From "Some cars have four wheels", the following constraints are extracted.

class(car), class(wheel), property(car,wheel:4)

Continuing with "A few cars have three wheels", we add

property(car,wheel:3)

and now the constraint solver should approach the two property constraints and change them into

property(car,wheel:3..4)

Continuing with "Other cars have 6 wheels" should add another constraint, and the constraint solver should combine the information and produce

property(car,wheel:3..6)

Example 3:
From "John is a man", the following constraints are extracted.

class(man), object(john,man).

Now, saying "John has a car" produces

property(john,car:1)

which the system hurries to convert to a constraint about the class rather than a particular object:

property(man,car:1)

Question 1 (warm up and refreshing Prolog)
Informing you that Prolog has declared ":" as an infix operator, op(550, xfy,:), can you explain the effect of the following declaration:

```
:- op(449,xfx,'..').
```

Question 2 (refresh CHR syntax)
What do the following declarations in CHR mean:

```
:- use_module(library(chr)).

handler knowledge_manager.

constraints     class/1,
                object/2,
                property/2.
```

Question 3
Write the Constraint Handling Rules that govern the collection of constraints of a class so that for there is only one constraint about a given property for given class; see example 1–3 above. You are welcome to apply the code fragments shown in questions 1 and 2 in your solution; test the solution by entering constraints as queries to Prolog (following "?-").

Question 4
Extend with a rule that implements the conversion of properties-of-objects into properties-of-classes, cf. example 3.

References

[1] Henning Christiansen, Christian Theil Have, Knut Tveitane. From use cases to UML class diagrams using logic grammars and constraints. *Proc. RANLP 2007, Recent Advances in Natural Language Processing; September 27-29, 2007, Borovets, Bulgaria*. 2007 (to appear).

[2] Henning Christiansen, Christian Theil Have, Knut Tveitane. Reasoning about Use Cases using Logic Grammars and Constraints. In [3]; pp. 40-52.

[3] Henning Christiansen, Jørgen Villadsen (eds.). Proceedings of the 4th International Workshop on Constraints and Language Processing, CSLP 2007. *Computer Science Research Report* 113, Roskilde University, 2007.