Meaning in Context
– constraint programming for natural language analysis

Henning Christiansen
Professor, PhD
Roskilde University, Computer Science
P.O.Box 260, DK-4000 Roskilde, Denmark
henning@ruc.dk, http://www.ruc.dk/henning

Joint work with Veronica Dahl, Simon Fraser University, Canada
Work on constraints in Roskilde

- CONTROL project, 2004–2007, funded by Danish SNF
  “Constraints for Robust Languages processing”
  HC, John Gallagher, Jørgen Villadsen & assoc. Veronica Dahl, Ph. Blache

- Attention to Constraint Handling Rules

- CHR Grammars (HC, TPLP 2005). Grammar notation on top of CHR.

- Aspects of NLP, extensions to Prolog & CHR, abduction (HC,VD)

- Analysis and optimization of CHR (HC, JG)

- Looking for practical applications, e.g. deaf peoples’ sign language

- CSLP workshop, Roskilde 2004. LNAI 3834; Barcelona 2005?
  *Int’l workshop on Constraint Solving and Language Processing*
Meaning in Context

An illustration of how CLP attitude may bring clarity into NLP.

- Simplified notion of meaning that fits better with pragmatics
- Basis for co-routining among different layers of analysis
- Each sentence interpreted and understood in context, and contributes to context

Technological background …
Technological background

- CHR, Constraint Handling Rules [Frühwirth, 1995, …]
  - declarative extension to Prolog for writing constraint solvers
  - available in SICStus Prolog (among others)

- A smart way of doing *abduction* in Logic Programming with CHR
  - efficient & light-weight
  - negation limited to so-called explicit negation
Constraint Handling Rules, intro. by example

:- use_module(library(chr)).
handler leq.
constraints leq/2.
:- op(500, xfx, leq).
X leq Y , Y leq Z ==> X leq Z.
X leq Y , Y leq X <=> X=Y.
X leq Y <=> X=Y | true.
X leq Y \ X leq Y <=> true.

p(X,Y):- q(X), r(Y,Z), X leq Z.
...

Execution model: Constraint store, replace/add constraints
Declarative semantics: as indicated by arrow symbols
Abduction in Logic Programming

Reasoning to find those “missing facts” of a Prolog program necessary to make given query succeed.

- Hot topic at logic programming conf. in 90ies

- Applications:
  - planning (event calculus and otherwise)
  - diagnosis
  - view updates in databases

- Typical implementations by meta-interp., \(\approx 20-100 \times\) slower than Prolog

An *abductive logic program* is a Prolog program extended with

- A set of distinguished, *abducible* predicates

- A set of *integrity constraints* to be satisfied by facts invented by interpreter
Abduction in Prolog + a little bit of CHR

Our trick:

- Abducibles $\rightarrow$ CHR constraints
- Integrity constraints $\rightarrow$ CHR rules
- Abductive programs run as Prolog, but with CHR taking care of abducibles

Example, view update in a database...
Abduction in Prolog + a little bit of CHR

Example, view update in a database

```
constraints father/2, mother/2, male/1, female/2.
parent(X,Y):- father(X,Y) ; mother(X,Y).
initial_db:- father(peter,john), male(peter).

mother(X,_) ==> female(X).
father(X,Z),father(Y,Z) ==> X=Y.
...

?- initial_db, parent(jane,john).
...
mother(jane,john),
female(jane) ?
```
Analysis of (natural) language as abduction

Given discourse assumed faithful to some “real world”

Context: (Partial) knowledge about this world

Basic assumption: Grammar ∨ Context → Sentences

Observe: This works fine with Prolog (DCG) to generate or verify given discourse known Context.

Discourse analysis is an abductive problem.

An example…
An example, analysis of discourses about still-lifes

Integrity constraints about abducible context facts:

\[\text{i\_on}(X,Y), \text{i\_on}(Y,X) \implies \text{fail}. \quad \text{container}(C) \implies \text{thing}(C).\]
\[\text{i\_in}(\text{the\_box}, \text{the\_vase}) \implies \text{fail}. \quad \text{i\_in}(\_, C) \implies \text{container}(C).\]
... \\
\text{thing}(X) \implies X=\text{the\_flower}; X=\text{the\_box}; X=\text{the\_vase}; X=\text{the\_table}. \\
\text{container}(X) \implies X=\text{the\_box}; X=\text{the\_vase}. \\
\text{container}(\text{the\_flower}) \implies \text{fail}. \quad \text{container}(\text{the\_table}) \implies \text{fail}. \\
\text{on}(X,Y) \implies \text{i\_on}(X,Y); \text{i\_on}(X,Z), \text{i\_on}(Z,Y); \text{i\_in}(X,Z), \text{i\_on}(Z,Y).

The grammar:

\[\text{sentence} \rightarrow [A, \text{is}, \text{on}, B], \{\text{thing}(A), \text{thing}(B), \text{on}(A,B)\}.\]

Query: ?- phrase(sentence, [the, flower, is, on, the, table])

One of the answers: \text{i\_in}(\text{the\_flower}, \text{the\_vase}), \text{i\_on}(\text{the\_vase}, \text{the\_table}), ...
Meaning-in-context: Taking pragmatics serious

*He won it*

The tall, red-haired man carrying a laptop won a brand new Ferrari

Both may have the same meaning or purpose to express: \(\text{won}(X, Z)\) where \(X\) and \(Y\) are references to objects in (presupposed) context:

\[
\text{tall}(X), \text{read_haired}(X), \text{carries}(X, Y), \text{laptop}(Y), \text{ferrari}(Z), \text{brand_new}(Z)
\]

Or to model a passive agent spying a discourse:

Sentence meaning: \(\emptyset\)

Sentence presupposes context:

\(\text{won}(X, Z), \text{tall}(X), \ldots\)

As opposed to std. Montague semantics with context-independent \(\lambda\)-terms ...
Perspectives

- Separating out context greatly simplifies semantic terms
- Seems more “pragmatics-oriented”
- Allows all levels of analysis interact with context,
  – resolve lexical ambiguities
  – identify and employ predefined contexts,
    gear-box ... brakes ... 200kmh ... ==> predef_context(all_about_cars)
- Approach is formalized by *possible-worlds semantics* [not shown today];
  formalization and generalization of R. Stalnaker’s theories about context
  and accommodation
Conclusion

- Abduction in Logic Programming can be implemented elegantly and efficiently in CHR
- Works with CHR Grammars [HC, 2001, ..., TPLP 2005] and Prolog in $A^2$LP paradigm [HC,VD, MultiCPL 2004]
- Constraint programming (CHR) revives abduction for NL interpretation
- ... and gives inspiration to Meaning-in-Context model