Constraint Programming: from LP to CLP

Eric MONFROY

IRIN, Université de Nantes

From Logic Programming
to
Constraint Logic Programming

Objectives

- historical integration of CP in LP
- reminder of LP
- main difference between LP and CLP (execution)

Brief historical review

- 1963. Sketchpad : introduction of CP techniques (Interactive drawing.)
- 1975. First formalism for CP techniques : scene labeling (recognition of 3D objects from 2D drawings)
- 1985. Discovering that logic programming is a special instance of constraint programming :

  unification = resolution of constraints over trees

  Moreover : Prolog is a relationnal and declarative language, and it offers features to explore the search space (backtracking)
  ⇒ Constraint Logic Programming (CLP)
- Constraints supply a relationnal aspect to Prolog arithmetic (vs. the `is/2` predicate)

Prolog syntax reminder (1)

- **Variables** : starts with an upper-case letter or underscore

  `X, _e34, Variable3`

- **Constants** : starts with a lower-case letter

  `a, 'character quotes', pi`

- **Structure** : `date(sunday,X,year(1999))`

- **Term** : variable, constant, or structure (= data structure of the program)

- **Atom** : expression of the form `p(t1,...,tn)` where `p` is a **predicate symbol** and the `ti`'s are terms

- **Fact** : expression of the form `p(t1,...,tn).`

  `father(john,X)`

Prolog syntax reminder (2)

- **Rule** : expression of the form :

  p(t,. . .,t) :- p(t,. . .,t), . . ., p(t,. . .,t).

  Example :

  ```
  menu(E,P,D) :- starter(E), main_dish(P), dessert(D).
  ```

- **Head** : left-hand side of the rule
- **Body** : right-hand side of the rule
- **Clause** : rule or fact
- **Query** : clause without head

Prolog syntax reminder (3)

- **Predicate** : set of clauses whose head have the same name and the same arity

  Example :

  ```
  dog(rex).
  dog(X) :- pet(X), bark(X).
  dog(X) :- bite(X).
  ```

- “,” : conjunction
- “ ;” : disjunction
- a fact describe a basic truth : « rex is a dog »
- meaning of the 2nd clause : « X is a dog if X is a pet and X barks ».
- each clause of a predicate is an alternative (implicit “or”)
- the name of a variable is local to the clause

Prolog execution

- procedural view of clauses, where call with parameter passing is replaced by call with variable unification.
- unification of terms : minimal substitution of variables that make the 2 terms equal.
- no description of the operationnal aspect : the order to consider clauses and goals is arbitrary (in theory).
- use of backtracking to explore all the alternatives

LP versus CLP (1)

- Prolog approach :

```
p(X,Y,Z):- Z is X+Y.
:- p(3,4,Z).
Z = 7
:- p(X,4,7).
INSTANTIATION ERROR
```

- CLP approach :

```
p(X,Y,Z):- Z #= X+Y.
:- p(3,4,Z).
Z = 7
:- p(X,4,7).
X = 3.
```

LP versus CLP (2)

Prolog arithmetic is not relationnal
⇒ the only possible approach is *generate and test* :

```
solution(X,Y,Z):- p(X), p(Y), p(Z), test(X,Y,Z).
p(11). p(3). p(7). p(16). p(15). p(14).
test(X,Y,Z):- Y is X+1, Z is Y+1.


:- solution(X,Y,Z).
```

458 steps to get the first solution

LP versus CLP (3)

CLP arithmetic is relationnal
⇒ possible approach : *constrain and generate*

```
solution(X,Y,Z):- test(X,Y,Z), p(X), p(Y), p(Z).
p(11). p(3). p(7). p(16). p(15). p(14).
test(X,Y,Z):- Y #= X+1, Z #= Y+1.
```

```
:- solution(X,Y,Z).
```

11 steps to get the first solution