Answer Set Programming via Controlled Natural Language Processing

Rolf Schwitter

Rolf.Schwitter@mq.edu.au

Overview

- Natural Language Processing
- Controlled Natural Language Processing
- Answer Set Programming
- ASP Methodology
- Reconstruction of a NL Specification in CNL
- Translation of CNL into ASP
- Reasoning in ASP
- Evaluation

The Paris Marathon Puzzle

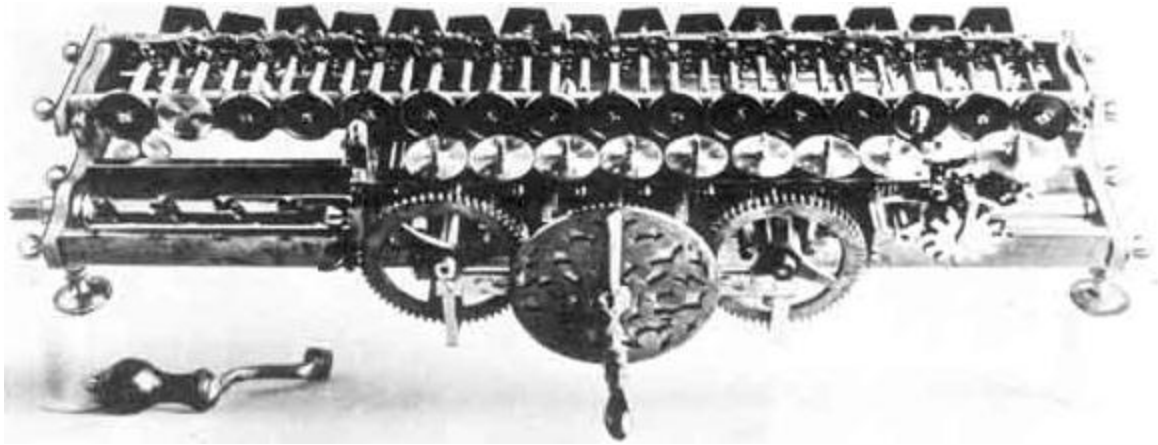
Dominique, Ignace, Naren, Olivier, Philippe, and Pascal have arrived as the first six at the Paris marathon.

Reconstruct their arrival order from the following information:

a) Olivier has not arrived last.
b) Dominique, Pascal and Ignace have arrived before Naren and Olivier.
c) Dominique who was third last year has improved this year.
d) Philippe is among the first four.
e) Igance has arrived neither in second nor third position.
f) Pascal has beaten Naren by three positions.
g) Neither Ignace nor Dominique are on the fourth position.

Goal

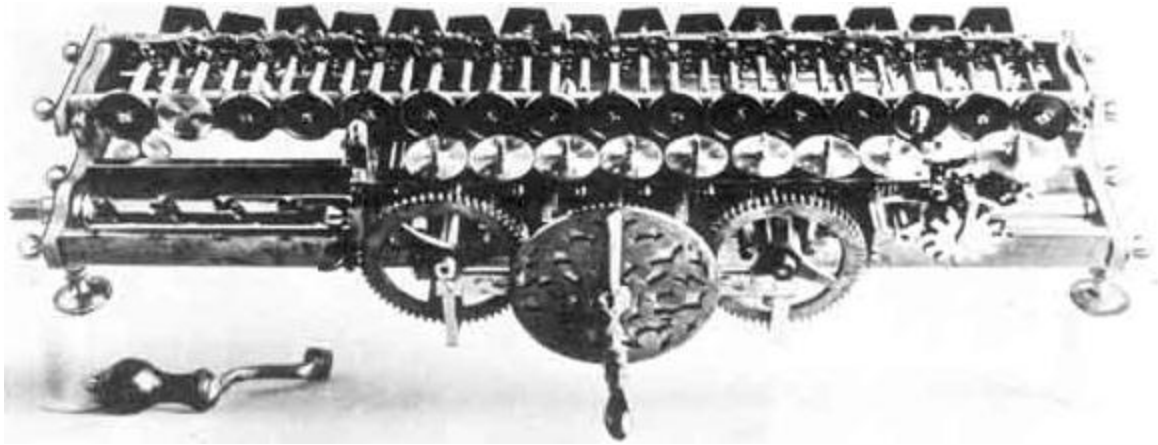
- Feed the puzzle to a machine:



in order to find the solution to the problem.

Goal

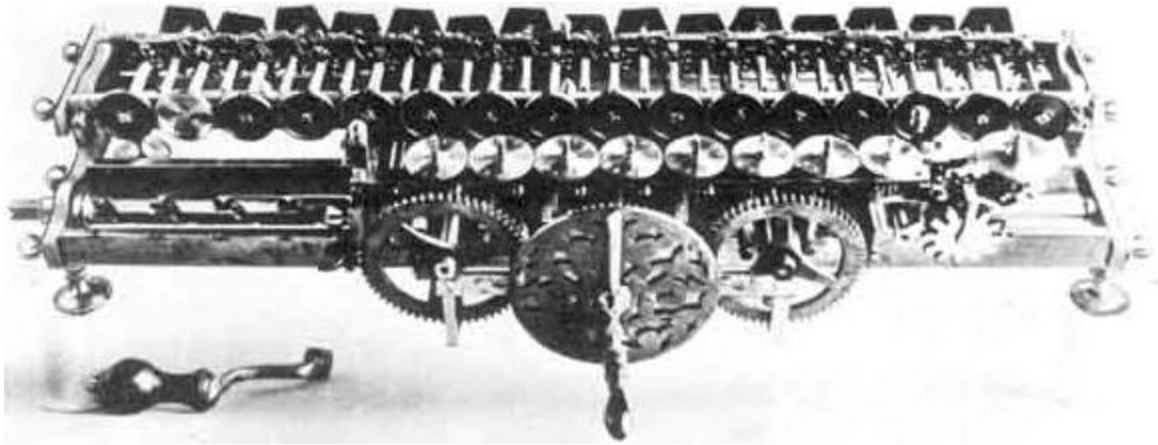
- Feed the puzzle in a machine-processable form to a machine:



in order to find the solution to the problem.

Goal

- Feed the puzzle in a human-readable and machine-processable form to a machine:



in order to find the solution to the problem.

Not Very Human-Readable ...

```
runner(dominique). runner(ignace). runner(naren). runner(olivier). runner(pascal). runner(philippe).
position(1..6).
1 { allocated_to(A,B) : position(B) } 1 :- runner(A).
:- runner(C), position(D), allocated_to(C,D), runner(E), allocated_to(E,D), C!=E.
:- allocated_to(olivier,6).
before(F,G) :- runner(F), position(H), allocated_to(F,H), runner(G), position(I), allocated_to(G,I), H<I.
:- before(naren,dominique).
:- before(naren,pascal).
:- before(naren,ignace).
:- before(olivier,dominique).
:- before(olivier,pascal).
:- before(olivier,ignace).
:- position(J), J>=3, allocated_to(dominique,J).
:- position(K), K>4, allocated_to(philippe,K).
:- allocated_to(ignace,2).
:- allocated_to(ignace,3).
:- position(L), allocated_to(pascal,L), position(M), allocated_to(naren,M), L!=M-3.
:- allocated_to(ignace,4).
:- allocated_to(dominique,4).
```

Not Very Machine-Processable ...

Dominique, Ignace, Naren, Olivier, Philippe, and Pascal have arrived as the first six at the Paris marathon.

Reconstruct their arrival order from the following information:

a) Olivier has not arrived last.
b) Dominique, Pascal and Ignace have arrived before Naren and Olivier.
c) Dominique who was third last year has improved this year.
d) Philippe is among the first four.
e) Igance has arrived neither in second nor third position.
f) Pascal has beaten Naren by three positions.
g) Neither Ignace nor Dominique are on the fourth position.

Not Very Machine-Processable ...

Dominique, Ignace, Naren, Olivier, Philippe, and Pascal have arrived as the first six at the Paris marathon.

Reconstruct their arrival order from the following information:

a) Olivier has not arrived last.
b) Dominique, Pascal and Ignace have arrived before Naren and Olivier.
c) Dominique who was third last year has improved this year.
d) Philippe is among the first four.
e) Igance has arrived neither in second nor third position.
f) Pascal has beaten Naren by three positions.
g) Neither Ignace nor Dominique are on the fourth position.

Towards a Solution

- Reconstruct the puzzle in controlled natural language.
- Translate the spec automatically into a formal language.
- Use Answer Set Programming (ASP) as formal language.
- Process the ASP program with an ASP solver.
- In our case, we use PENG Light as CNL.
- CNL -> DRS -> ASP notation -> Clingo (ASP solver).
- Important: get the conceptualisation right.
- Important: writing support (predictive editor).

Answer Set Programming (ASP)

- ASP has its roots in knowledge representation, logic programming and constraint satisfaction.
- Notation similar to Prolog but uses an entirely different computational mechanism.
- Finding a solution in ASP corresponds to computing stable models (= answer sets).

Answer Set Programs

- Basic rules (similar to Prolog)
- Disjunctive rules (to express logical uncertainty)
- Choice rules (enumerate potential solutions)
  - choice rules plus cardinality constraints
  - choice rules with conditional literals
- Constraints (for excluding solutions)
- Directives (for displaying partial answer sets)

ASP Methodology

- Define predicates (with basic rules).
- Generate potential solutions (with choice rules).
- Weed out solutions (with constraints).
- Display selected solutions (with directives).

Conceptualisation

- Conceptualisation is important for the reconstruction of the puzzle.
- Keep in mind, we have:

  runners: Dominique, Ignace, Naren, ...

  positions: 1..6

  runners are allocated to positions

Reconstruction in PENG Light

- Dominique, Ignace, Naren, Olivier, Philippe, and Pascal have arrived as the first six at the Paris marathon.
- Dominique, Ignace, Naren, Olivier, Philippe and Pascal are runners.
- There exist exactly six positions.

Reconstruction in PENG Light

- Dominique, Ignace, Naren, Olivier, Philippe, and Pascal have arrived as the first six at the Paris marathon.

- Every runner is allocated to exactly one position.
- Reject that a runner R1 is allocated to a position and that another runner R2 is allocated to the same position and that R1 is not equal to R2.

Reconstruction in PENG Light

- Olivier has not arrived last.
- Reject that Olivier is allocated to the sixth position.

Reconstruction in PENG Light

- Dominique, Pascal and Ignace have arrived before Naren and Olivier.
- If a runner R1 is allocated to a position P1 and another runner R2 is allocated to a position P2 and P1 is smaller than P2 then R1 is before R2.
- Reject that Narren is before Dominique, Pascal, and Ignace.
- Reject that Olivier is before Dominique, Pascal, and Ignace.

Reconstruction in PENG Light

- Dominique who was third last year has improved this year.
- Reject that Dominique is allocated to a position that is greater than or equal to 3.

Reconstruction in PENG Light

- Philippe is among the first four.
- Reject that Philippe is allocated to a position that is greater than 4.

Reconstruction in PENG Light

- Ignace has arrived neither in second nor third position.

- Reject that Ignace is allocated to the second position.
- Reject that Ignace is allocated to the third position.

Reconstruction in PENG Light

- Pascal has beaten Naren by three positions.
- Reject that Pascal is allocated to a position P1 and that Naren is allocated to a position P2 and that P1 is not equal to P2 minus 3.

Reconstruction in PENG Light

- Neither Ignace nor Dominique are on the fourth position.

- Reject that Ignace is allocated to the fourth position.
- Reject that Dominique is allocated to the fourth position.

Translation into DRS

```
[A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,Q,R,S,T]
named(A,dominique)                          [V]
object(B,runner)                            object(V,runner)
predicate(C,isa,A,B)                        ==>
named(D,ignace)                                [W,X,Y]
object(E,runner)                               cardinal(W,eq,1)
predicate(F,isa,D,E)                           object(W,position)
named(G,naren)                                 predicate(X,be,V)
object(H,runner)                               property(X,allocated_to,W)
predicate(I,isa,G,H)                        NOTc:
named(J,olivier)                               [Z,A1,B1,C1,D1,E1,F1,G1]
object(K,runner)                               object(Z,runner)
predicate(L,isa,J,K)                           variable(Z,r1)
named(M,pascal)                                object(A1,position)
object(N,runner)                               predicate(B1,be,Z)
predicate(O,isa,M,N)                           property(B1,allocated_to,A1)
named(P,philippe)                              object(D1,runner)
object(Q,runner)                               variable(D1,r2)
predicate(R,isa,U,Q)                           predicate(E1,be,D1)
cardinal(S,eq,6)                               property(E1,allocated_to,A1)
object(S,position)                             predicate(G1,be,Z)
predicate(T,exist,S)                           operator(G1,\=,D1)
```

Translation into ASP

- Dominique, Ignace, Naren, Olivier, Philippe, and Pascal are runners.

```
runner(dominique).
runner(ignace).
runner(olivier).
runner(philippe).
runner(pascal).
```

Translation into ASP

- There are exactly six positions.

```
position(1..6).
```

Translation into ASP

- Every runner is allocated to exactly one position.

```
1 { allocated_to(A,B) : position(B) } 1 :-
    runner(A).
```

Translation into ASP

- Reject that a runner R1 is allocated to a position and that another runner R2 is allocated to the same position and that R1 is not equal to R2.

```
:- runner(C),
   position(D),
   allocated_to(C,D),
   runner(E),
   allocated_to(E,D),
   C != E.
```

Translation into ASP

- Reject that Olivier is allocated to the sixth position.

```
:- allocated_to(olivier,6).
```

Translation into ASP

- If a runner R1 is allocated to a position P1 and another runner R2 is allocated to a position P2 and P1 is smaller than P2 then R1 is before R2.

```
before(G,H) :-
   runner(G),
   position(I),
   allocated_to(G,I),
   runner(H),
   position(J),
   allocated_to(H,J),
   I < J.
```

Translation into ASP

- Reject that Narren is before Dominique, Pascal, and Ignace.

```
:- before(naren,dominique).
:- before(naren,pascal).
:- before(naren,ignace).
```

Translation into ASP

- Reject that Olivier is before Dominique, Pascal, and Ignace.

```
:- before(olivier,dominique).
:- before(olivier,pascal).
:- before(olivier,ignace).
```

Translation into ASP

- Reject that Dominique is allocated to a position that is greater than or equal to 3.

```
:- position(K),
   K >= 3,
   allocated_to(dominique,K).
```

Translation into ASP

- Reject that Philippe is allocated to a position that is greater than 4.

```
:- position(L),
   L > 4,
   allocated_to(philippe,L).
```

Translation into ASP

- Reject that Ignace is allocated to the second position.

```
:- allocated_to(ignace,2).
```

- Reject that Ignace is allocated to the third position.

```
:- allocated_to(ignace,3).
```

Translation into ASP

- Reject that Pascal is allocated to a position P1 and that Naren is allocated to a position P2 and that P1 is not equal to P2 minus 3.

```
:- position(O),
   allocated_to(pascal,O),
   position(P),
   allocated_to(naren,P),
   O != P - 3.
```

Translation into ASP

- Reject that Ignace is allocated to the fourth position.

```
:- allocated_to(ignace,4).
```

- Reject that Dominique is allocated to the fourth position.

```
:- allocated_to(dominique,4).
```

Evaluation

- Clingo is a state-of-the-art answer set solver.
- Clingo generates and displays the following answer set if we use the directives:

```
#hide. #show allocated_to(A,B).

Answer 1:
allocated_to(ignace,1)
allocated_to(dominique,2)
allocated_to(pascal,3)
allocated_to(philippe,4)
allocated_to(olivier,5)
allocated_to(naren,6)
```

Evaluation

1. Dominique, Ignace, Naren, Olivier, Pascal, and Philippe are runners.
2. There exist exactly six positions.
3. Every runner is allocated to exactly one position.
4. Reject that a runner R1 is allocated to a position and that another runner R2 is allocated to the same position and that R1 is not equal to R2.
5. Reject that Olivier is allocated to the sixth position.
6. If a runner R1 is allocated to a position P1 and another runner R2 is allocated to a position P2 and P1 is smaller than P2 then R1 is before R2.
7. Reject that Naren is before Dominique, Pascal, and Ignace.
8. Reject that Olivier is before Dominique, Pascal, and Ignace.
9. Reject that Dominique is allocated to a position that is greater than or equal to 3.
10. Reject that Philippe is allocated to a position that is greater than 4.
11. Reject that Ignace is allocated to the second position.
12. Reject that Ignace is allocated to the third position.
13. Reject that Pascal is allocated to a position P1 and that Naren is allocated to a position P2 and that P1 is not equal to P2 minus 3.
14. Reject that Ignace is allocated to the fourth position.
15. Reject that Dominique is allocated to the fourth position.

Sentence No.	Answer Sets
1^c	1
2^c	1
3^c	46656
4^c	720
5^c	600
6^c	600
7^c	150
8^c	42
9^c	24
10^c	12
11^c	10
12^c	6
13^c	3
14^c	1
15^c	1

Question Answering

- We can add questions directly to our theory.
- For example:

  Which runner is allocated to what position?

  translates into:

```
answer(S,T) :-
   runner(T),
   position(S),
   allocated_to(T,S).
```

Take Home Messages

- Combinatory puzzles can be expressed in CNL and translated automatically into ASP programs.
- But we have to stick to the ASP methodology:
  - Generate -> Define -> Test -> Display
- Choice rules can be expressed as universally quantified sentences with cardinality restrictions.
- Predicates can be defined via conditional sentences.
- Constraints can be expressed as negated sentences.