Inductive Logic Programming Using a MaxSAT Solver

National Institute of Technology, Tokuyama College
Noriaki Chikara

Kyushu University
Miyuki Koshimura, Hiroshi Fujita, Ryuzo Hasegawa

Table of Contents

Introduction

- Inductive Logic Programming (ILP)
  - A method of inductive learning
  - ILP systems : Progol, Aleph, etc.
  - A wide variety of applications
- MaxSAT
  - Optimization version of Satisfiability Testing (SAT)
  - Satisfying clauses as many as possible.

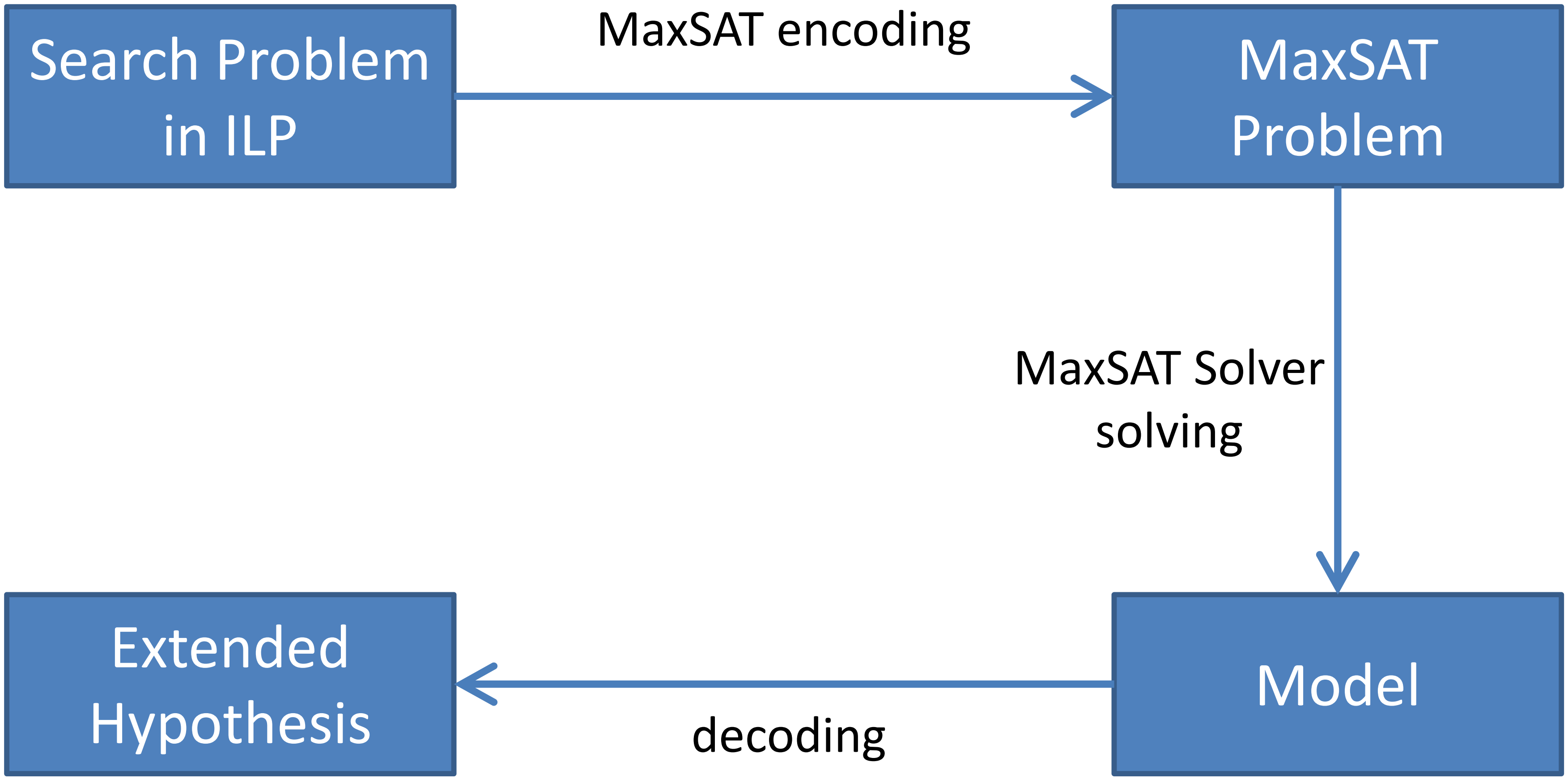
  - A lot of progress in SAT/MaxSAT solvers
  - Remarkable success of many applications

ILP using a MaxSAT solver

Our motivation is to increase the performance by using the MaxSAT solver.

- We propose a method which transforms a problem of ILP into that of MaxSAT.
- Mimicing the search in Progol and Aleph.
- A syntactical restriction on the ILP problem.
- Preprocessing in order to prevent the size of the transformed problem growing up.

ILP using a MaxSAT Solver



Inductive Logic Programming (ILP)

Given the background knowledge B,

a set of positive examples E^+, and a set of negative examples E^-,

which satisfy the following relations:

$$\begin{cases} B \not\models E^+ \\ B \cup E^- \not\models \Box \end{cases}$$

an ILP system will derive a hypothesized logic program H

which satisfies the following relations:

$$\begin{cases} B \cup H \models E^+ \\ H \cup B \cup E^- \not\models \Box \end{cases}$$

Cover set algorithm (The algorithm of Aleph and Progol)

B is the background knowledge, H is hypotheses, and E is a set of positive examples. H is initialized to ∅.

(1) If $E = \emptyset$ then output H.

(2) Let e be an example in E.

(3) Generate a MSH from e and B.

Using MaxSAT solver

(4) Generate the best hypothesis H' with a **top-down search**.

(5) $H := H \cup H'$.

(6) $E' := \{e' | e' \in E \ and \ B \cup H' \vDash e'\}$.

(7) Goto (1).

(MSH: Most Specific Hypothesis)

Restricting ILP in this study

- Arguments in the predicate do not have structure.
- All predicates are required to having mode declarations
- We do not deal with negated atoms.

Preprocessing
Tree Structure of Hypothesis

Refinement operation of Hypothesis

head(A).

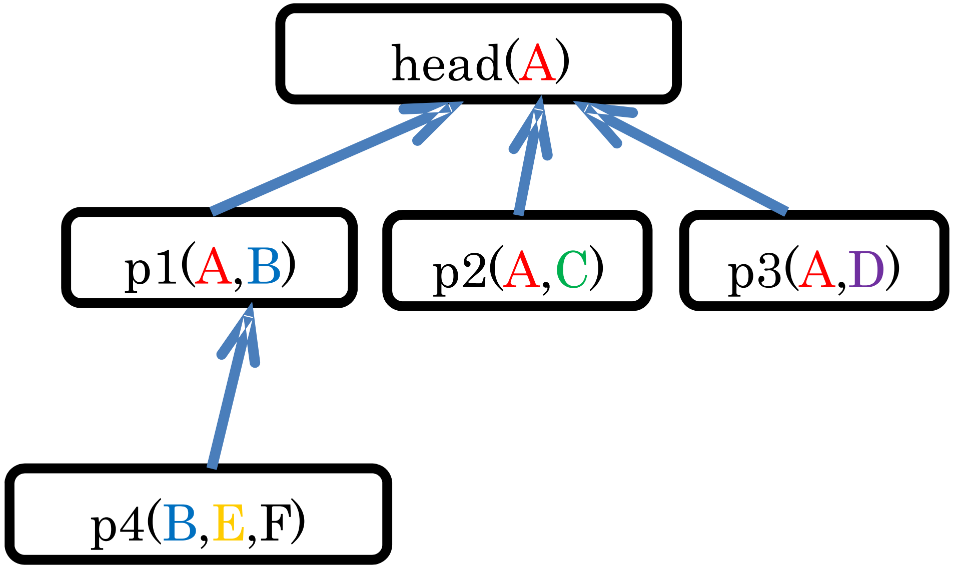
⇓ refinement

head(A):-p1(A,B),p2(A,C),p3(A,D).

⇓ refinement

head(A):-p1(A,B),p2(A,C),p3(A,D),p4(B,E,F).

Causality graph of literals in the hypothesis

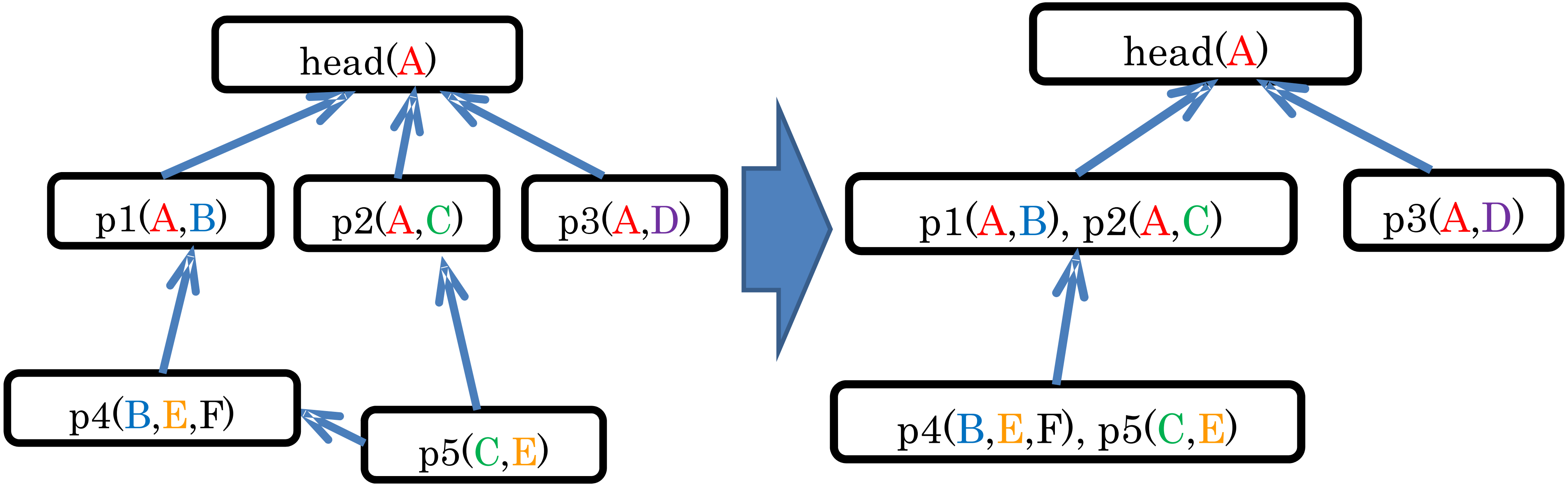


We have only to take care of its descendant nodes locally.
→Suppressing the size of MaxSAT encoding

The Transformation to Tree structure

Hypothesis :

head(A):-p1(A,B),p2(A,C),p3(A,D),p4(B,E,F),p5(C,E).



MaxSAT encoding

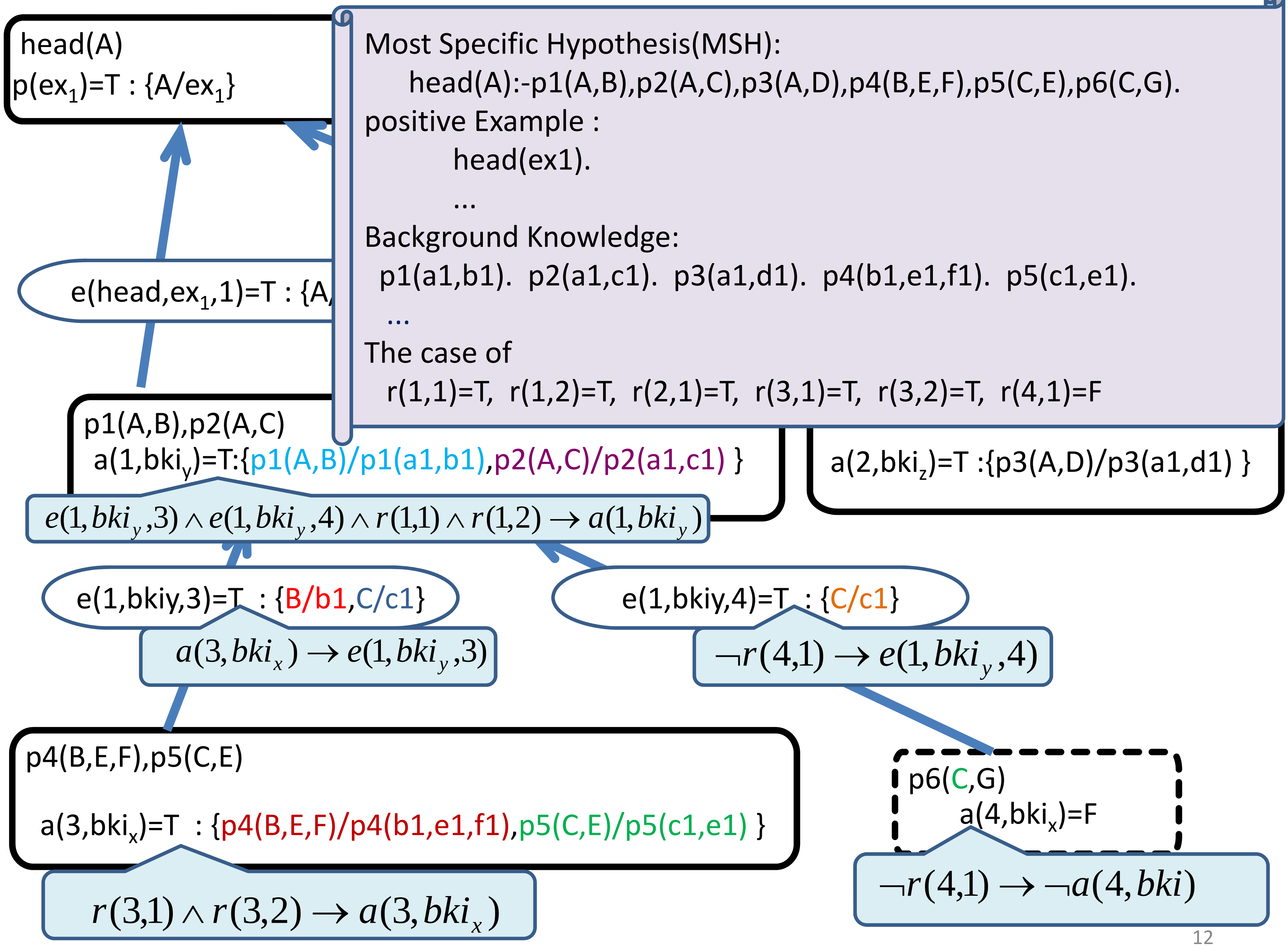
- *r*(*i; j*) : the *j*-th literal of the *i*-th literal group in *MSH* appears in *H′*.
- *a*(*i; bki*) : the *i*-th literal group in *MSH* is satised. Through *bki*, we can nd out ground unit clauses in the background knowledge which are used for the satisfaction.
- *e*(*ia; bkia; ib*) : the necessary condition for *a*(*ia; bkia*) is satised in the decendant nodes of the *ib*-th literal group in *MSH*.
- *p*(*ex*): *H′* covers the *ex*-th positive example.
- *n*(*ey*) : *H′* covers the *ey*-th negative example.

Soft Clauses:

-r(i,j). {weight:1}

p(ex). {weight: (number of body literal in MSH)+1}

(MSH: Most Specific Hypothesis)


head(A)
$p(ex_1)$=T : {A/ex_1}
Most Specific Hypothesis(MSH):
head(A):-p1(A,B),p2(A,C),p3(A,D),p4(B,E,F),p5(C,E),p6(C,G).
positive Example :
head(ex1).
...
Background Knowledge:
p1(a1,b1). p2(a1,c1). p3(a1,d1). p4(b1,e1,f1). p5(c1,e1).
...
The case of
r(1,1)=T, r(1,2)=T, r(2,1)=T, r(3,1)=T, r(3,2)=T, r(4,1)=F
e(head,ex_1,1)=T : {A/
p1(A,B),p2(A,C)
a(1,bki_y)=T:{p1(A,B)/p1(a1,b1),p2(A,C)/p2(a1,c1) }
a(2,bki_z)=T :{p3(A,D)/p3(a1,d1) }
$e(1,bki_y,3) \wedge e(1,bki_y,4) \wedge r(1,1) \wedge r(1,2) \rightarrow a(1,bki_y)$
e(1,bkiy,3)=T : {B/b1,C/c1}
e(1,bkiy,4)=T : {C/c1}
$a(3,bki_x) \rightarrow e(1,bki_y,3)$
$\neg r(4,1) \rightarrow e(1,bki_y,4)$
p4(B,E,F),p5(C,E)
a(3,bki_x)=T : {p4(B,E,F)/p4(b1,e1,f1),p5(C,E)/p5(c1,e1) }
p6(C,G)
a(4,bki_x)=F
$r(3,1) \wedge r(3,2) \rightarrow a(3,bki_x)$
$\neg r(4,1) \rightarrow \neg a(4,bki)$

Data set of UCI for the Experiment

- **Connect-4**

  A two-player connection game. This database contains all legal 8-ply positions of the game.

- **Audiology(Standardized)**

  Nominal audiology dataset from Baylor.

- **Molecular Biology(Splice-junction Gene Sequences)**

  Arimate splice-junction gene sequences (DNA) with associated imperfect domain theory.

A Environment of the Experiment

MaxSAT Solver：QMaxSAT14.04
based on Glucose3.0, on Cygwin1.7.28(64bit)

pre-processing and post-processing :Java
java 1.8.40(64bit)
Paser uses JavaCC

PC for Experiments:
OS:Windows7 (64bit)
CPU：Core i7-2620M(2.6GHz)
Memory:8GHz
SSD

The execution time of the experiment

Problem	Target of positive examples	Num. of extracted rules	Max. length of extracted rules	Our method (using MaxSAT)	Aleph+ YAProlog
Connect-4	win	7	5 literals	1.501s	9.360s
	draw	1	7 literals	0.877s	56.566s
	loss	1	4 literals	0.970s	0.640s
Audiology (Standardized)	cochlear age	2	6 literals	2.624s	1m39.169s
	cochlear age and noise	1	4 literals	2.530s	0.218s
	cochlear unknown	4	6 literals	3.934s	10m47.807s
	mixed cochlear age otitis media	1	3 literals	1.672s	0.187s
	possible menieres	1	2 literals	1.438s	0.172s
Molecular Biology(Splice-junction Gene Sequences)	EI	4	5 literals	4.418s	0.328s
	Neither	14	4 literals	1m4.166s	0.905s

Conclusion

- We proposed a new method that transforms a problem of ILP into that of MaxSAT.
- We converted an ILP problem to a tree structure, thereby we suppressed the size of MaxSAT encoding.
- Experimental results show that Our method works fairly well.

Future works :

- We apply the method to other ILP problems and to evaluate the performance.
- We speed up I/O of MaxSAT instances.