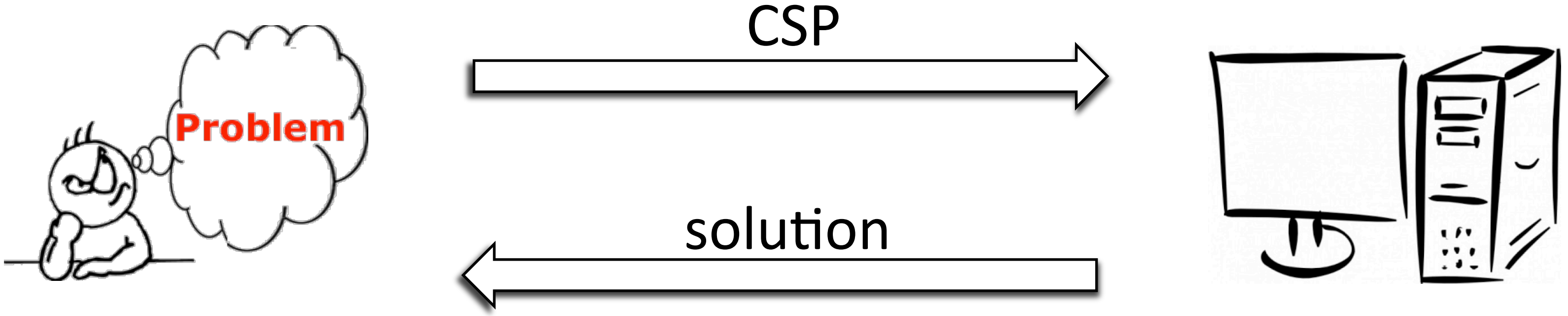
Boosting Constraint Acquisition via Generalization Queries

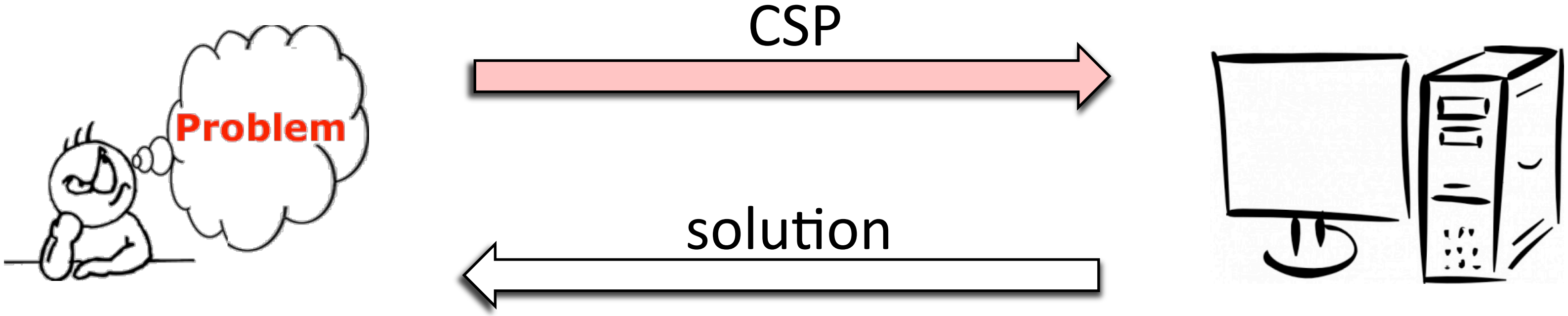
Christian Bessiere[1]
Nadjib Lazaar[1]
Remi Coletta[1]
Younes Mechqrane[1,2]
Abderrazak Daoudi[1,2]
El Houssine Bouyakhf[2]

[1]University of Montpellier, France
[2] LIMIARF/FSR, University Mohammed V Agdal, Rabat, Morocco

Context



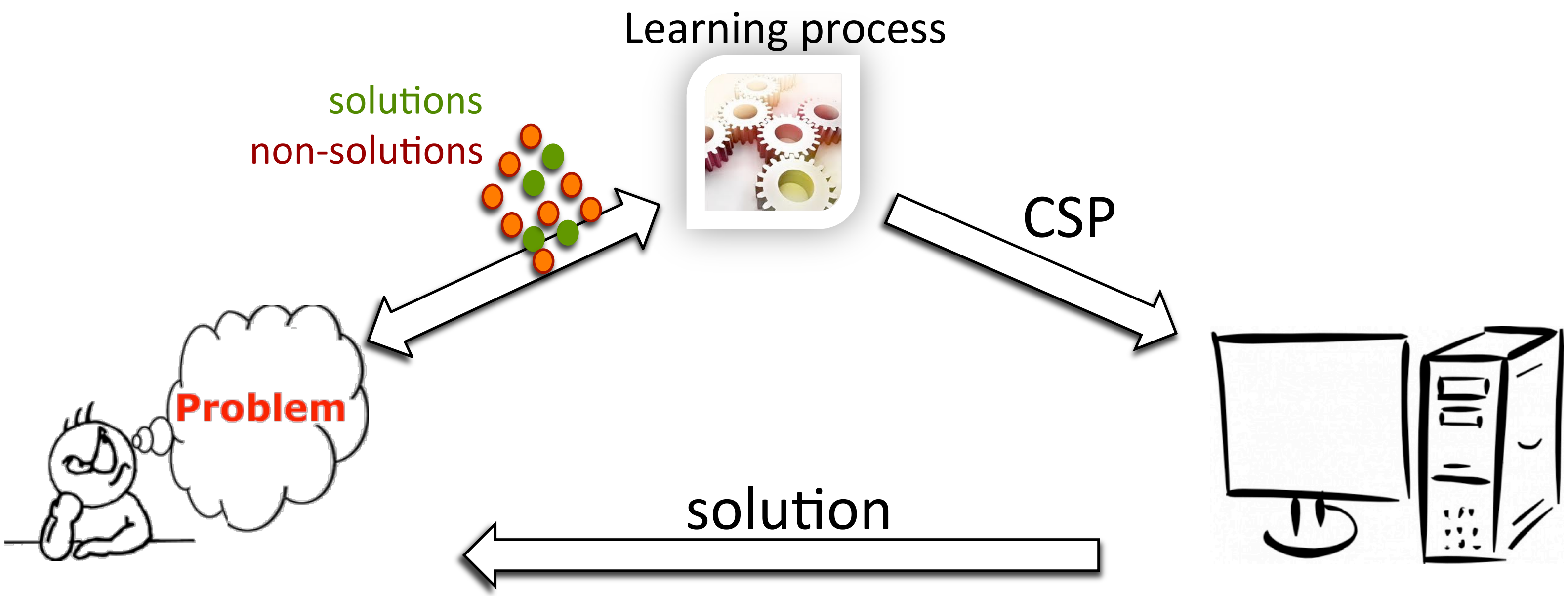
Context



- Question: How does the user write down the constraints of a problem?
- Limitations: modelling constraint networks requires a fair expertise

- Need: Simple way to build constraint models ➔ Modeller-assistant

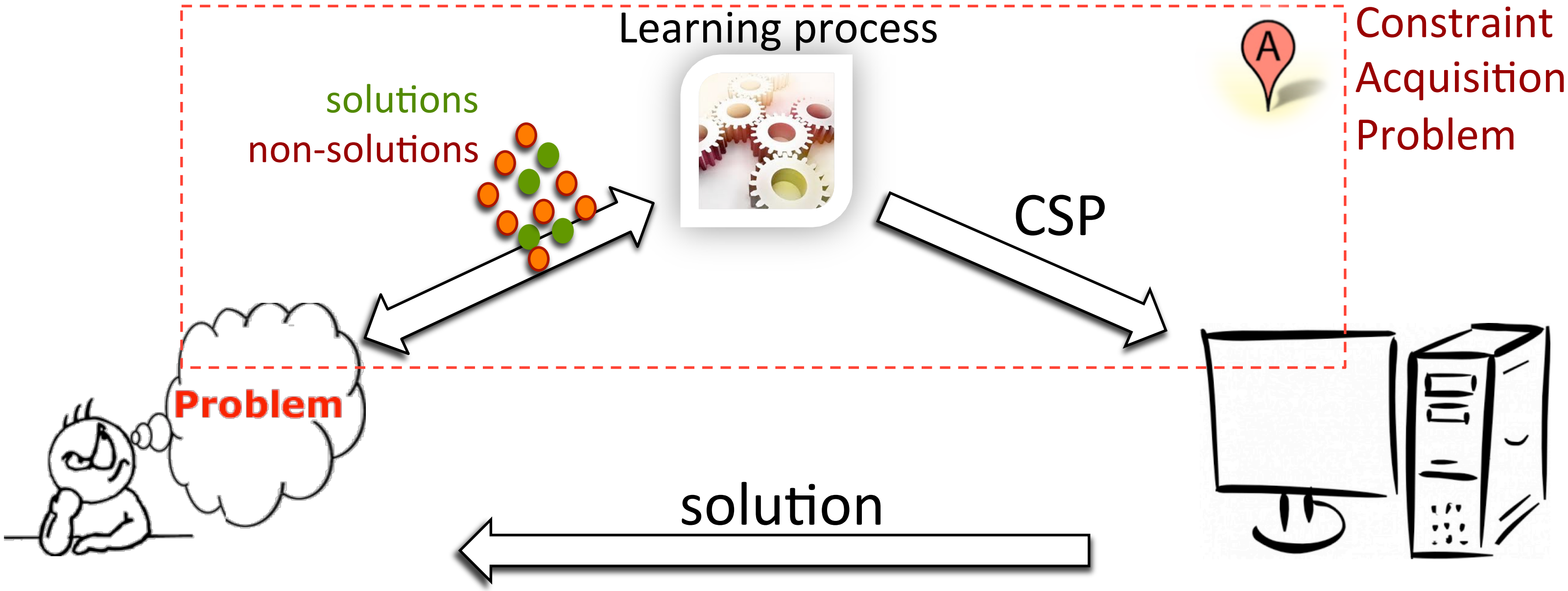
Context



- Question: How does the user write down the constraints of a problem?
- Limitations: modelling constraint networks requires a fair expertise

- Need: Simple way to build constraint models ➔ Modeller-assistant
- How: In a Machine Learning way (passive/active, offline/online, by reinforcement…)

Context



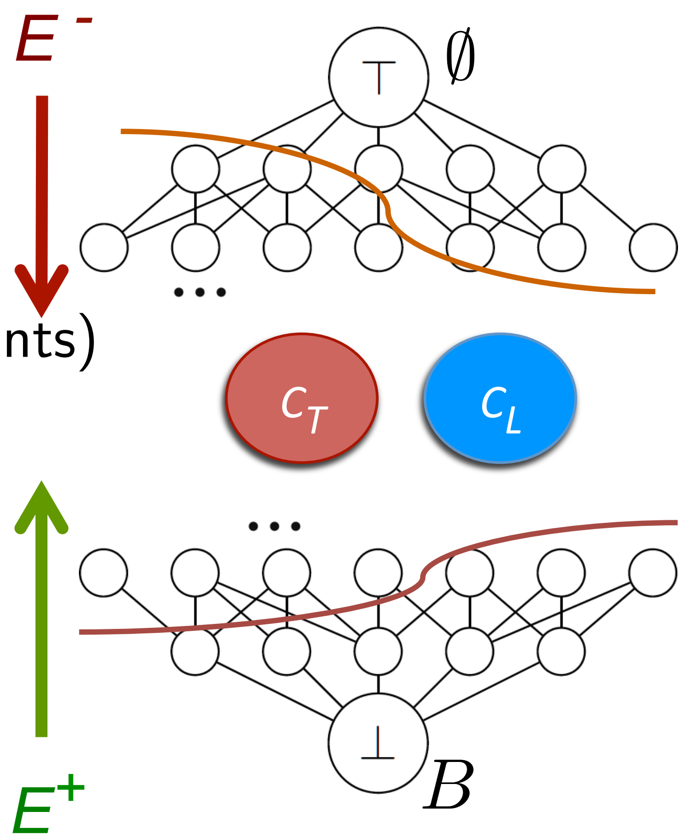
- Question: How does the user write down the constraints of a problem?
- Limitations: modelling constraint networks requires a fair expertise

- Need: Simple way to build constraint models ➔ Modeller-assistant
- How: In a Machine Learning way (passive/active, offline/online, by reinforcement…)

Constraint Acquisition Problem

Inputs:

- (X, D) : Vocabulary
- B : Basis (version space/possible constraints)
- C_T : Target Network
- (E^+, E^-) : positives and negatives



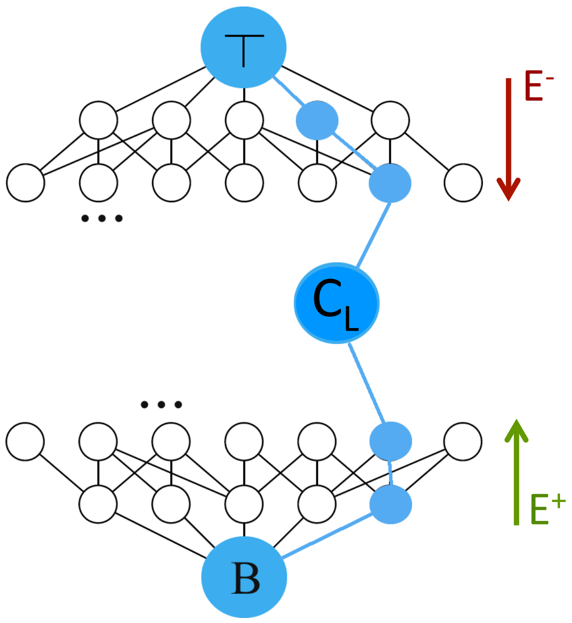
Output:

- C_L : Learned network such that:

$$C_L \subset B : C_L \equiv C_T$$

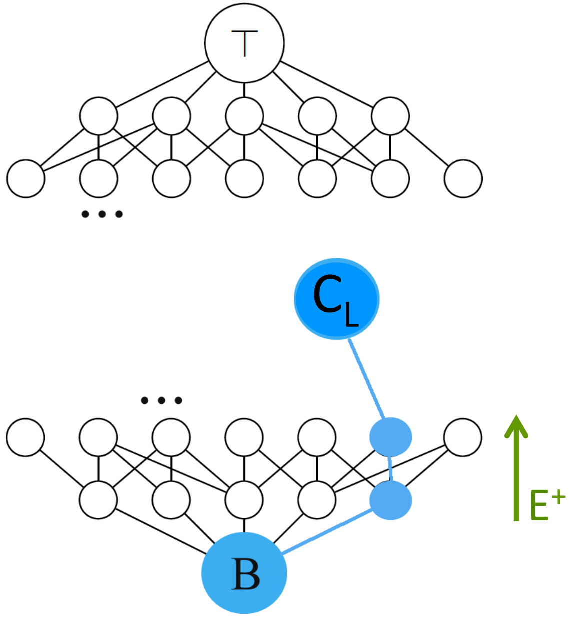
Constraint Acquisition Systems

- CONACQ
  - Conacq1.0 (passive learning) [Bessiere et al. ECML05]
  - Conacq2.0 (active learning) [Bessiere et al. IJCAI07]



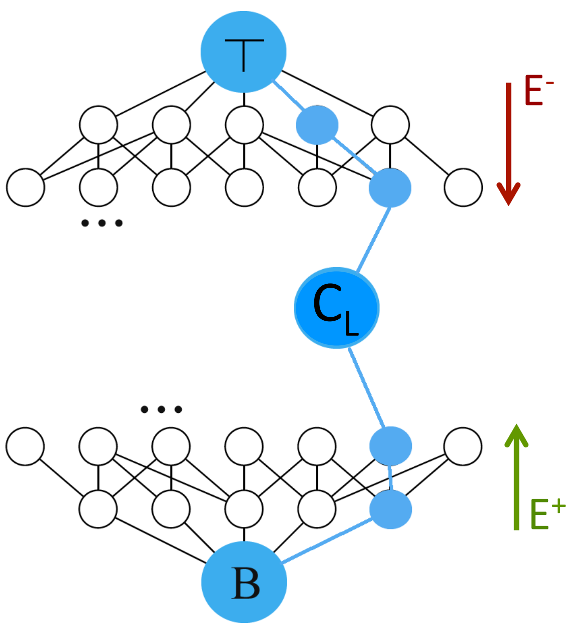
Constraint Acquisition Systems

- CONACQ
  - Conacq1.0 (passive learning) [Bessiere et al. ECML05]
  - Conacq2.0 (active learning) [Bessiere et al. IJCAI07]
- ModelSeeker [Beldiceanu and Simonis, CP12]
  - A passive learning
  - Based on global constraint catalog (more than 400)
  - Buttom-up search

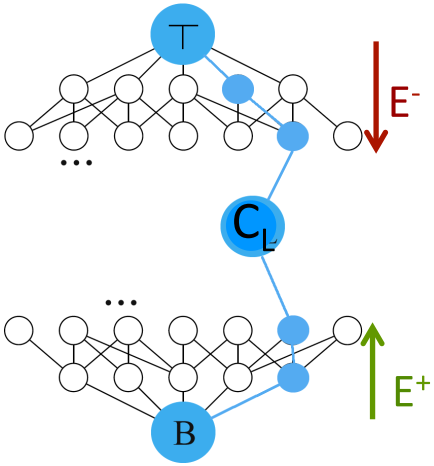


Constraint Acquisition Systems

- CONACQ
  - Conacq1.0 (passive learning) [Bessiere et al. ECML05]
  - Conacq2.0 (active learning) [Bessiere et al. IJCAI07]
- ModelSeeker [Beldiceanu and Simonis, CP12]
  - A passive learning
  - Based on global constraint catalog (more than 400)
  - Buttom-up search
- QUACQ [Bessiere et al. IJCAI13]
  - Active learning approach
  - Based on partial queries to elucidate the scope of the constraint to learn



Motivations



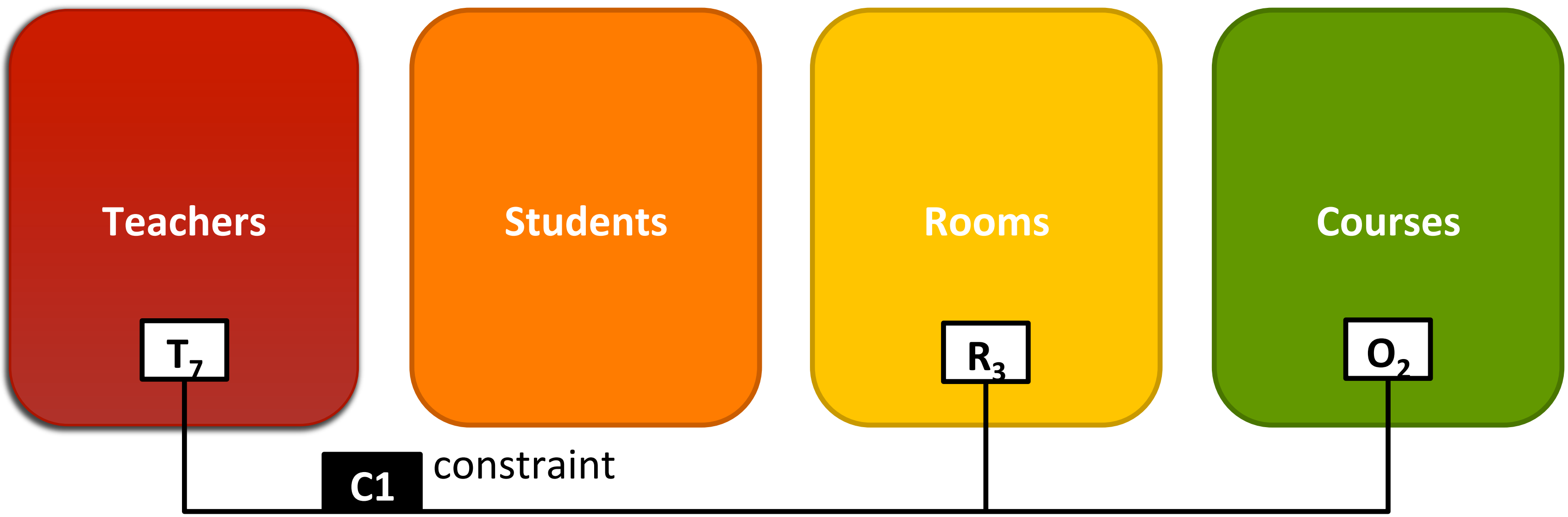
Limitation:
- Hard to put in practice:
  - ModelSeeker: cannot learn on unstructured problems
  - CONACQ and QUACQ: more than **8000** queries to learn the Sudoku model

Need:
- Reduce the dialogue with the user to make constraint acquisition more efficient in practice

How:
- Eliciting more information by asking complex queries to the user

Variables and Types

- A type is a subset of variables defined by the user as having a common property
- Example (School Timetabling Problem)



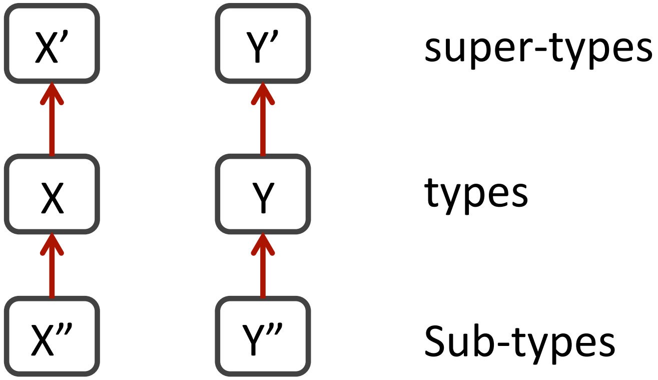
Can C1 be generalized to all Teachers, Rooms and Courses?

Generalization Query

- Let $c(x,y)$ a learned constraint and X,Y are types of x,y :
  - Generalization Query: $AskGen((X,Y),c)$
- The user says yes iff the constraint c holds on all possible scope

$$(x_i, y_i) \in (X,Y)$$

- Properties

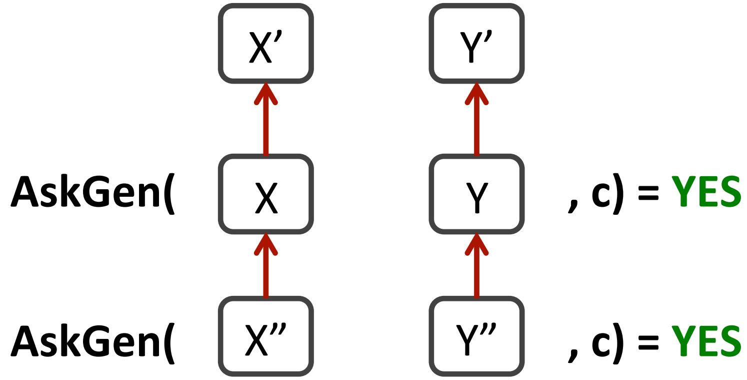


Generalization Query

- Let $c(x, y)$ a learned constraint and X, Y are types of x, y :
  - Generalization Query: $AskGen((X,Y),c)$
- The user says yes iff the constraint c holds on all possible scope

$$(x_i, y_i) \in (X, Y)$$

- Properties

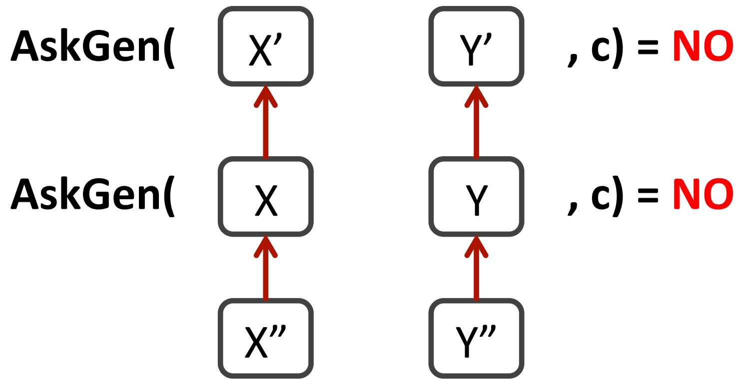


Generalization Query

- Let $c(x, y)$ a learned constraint and X, Y are types of x, y :
  - Generalization Query: $AskGen((X, Y), c)$
- The user says yes iff the constraint c holds on all possible scope

$$(x_i, y_i) \in (X, Y)$$

- Properties



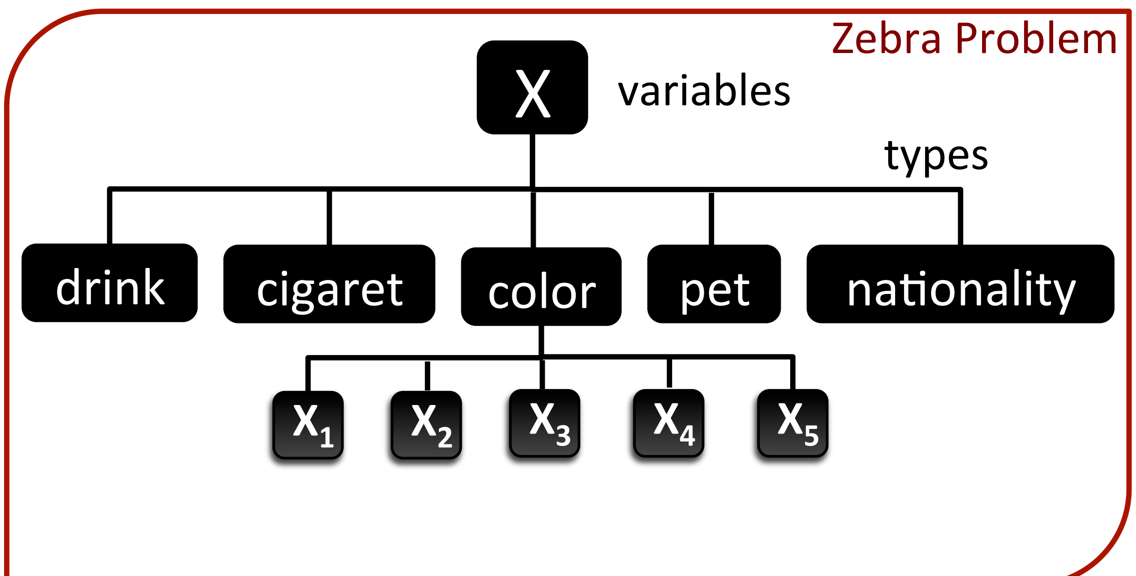
GENACQ

Inputs

- A learned constraint
- Combination of possible types (i.e., table)

Output

- Set of constraints



GENACQ

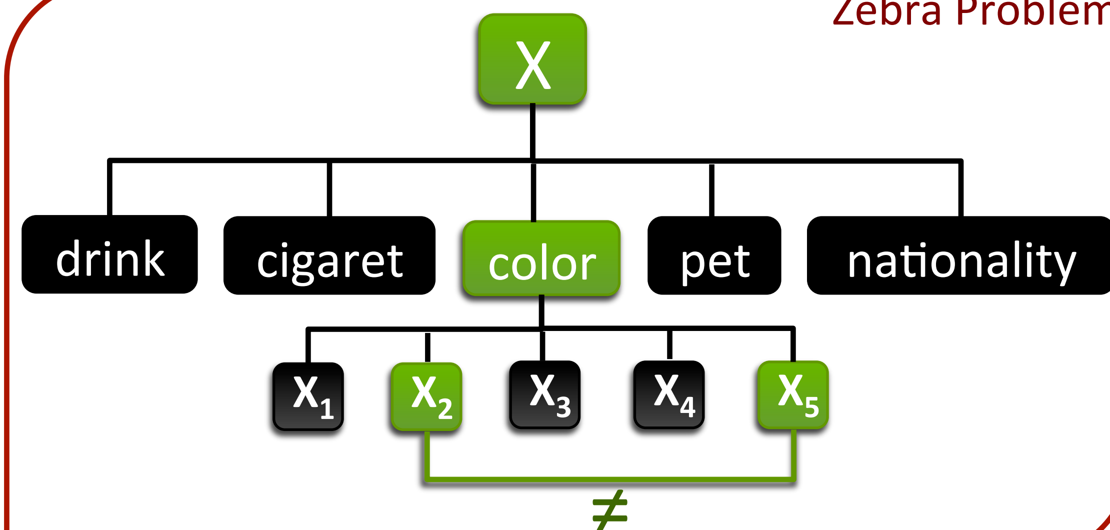
Inputs

- A learned constraint
- Combination of possible types (i.e., table)

Output

- Set of constraints

Zebra Problem



INPUTS

- Learned constraint : $x_2 \neq x_5$
- Table:

#q = 4

askGen

x_2	x_5	
x_2	color	✓
x_2	X	✗
color	x_5	✓
color	color	✓
color	X	✗
X	x_5	✗
X	color	✗
X	X	✗

GENACQ

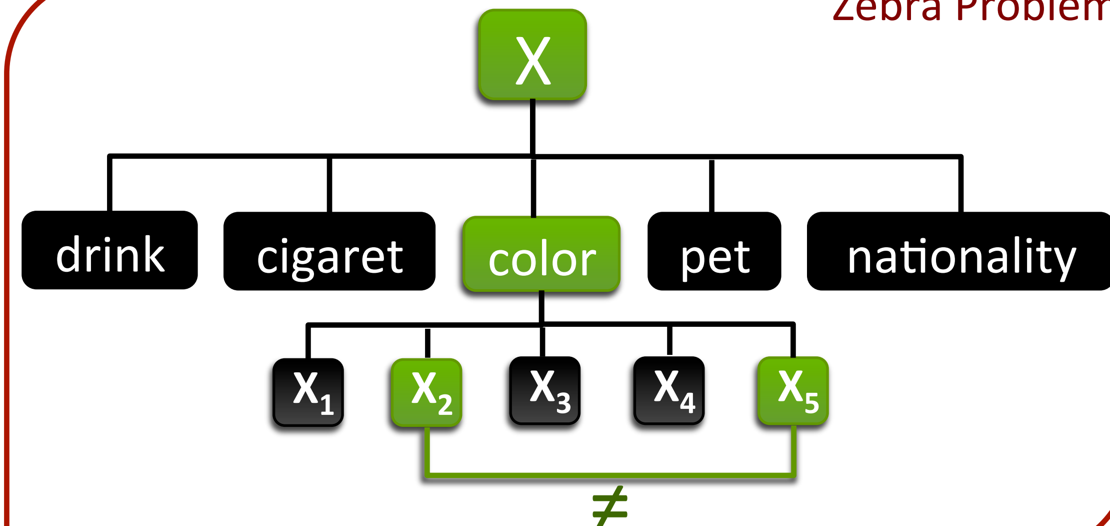
Inputs

- A learned constraint
- Combination of possible types (i.e., table)

Output

- Set of constraints

Zebra Problem



INPUTS

- Learned constraint : $x_2 \neq x_5$
- Table:

#q = 5

x_2	x_5	
x_2	color	✓
x_2	X	✗
color	x_5	✓
color	color	✓
color	X	✗
X	x_5	✗
X	color	✗
X	X	✗

GENACQ

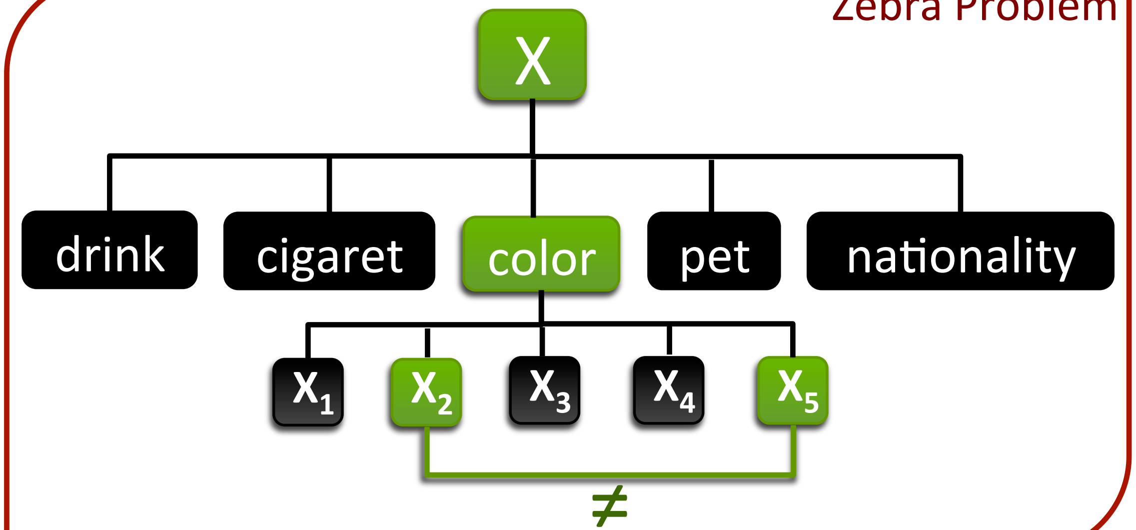
Inputs

- A learned constraint
- Combination of possible types (i.e., table)

Output

- Set of constraints

Zebra Problem

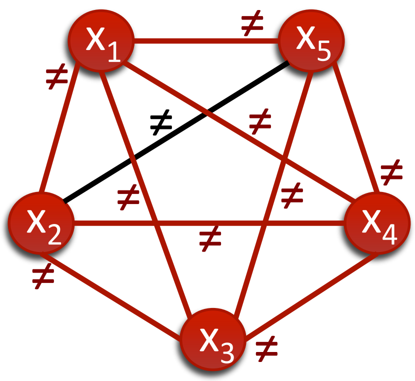


INPUTS

- Learned constraint : $x_2 \neq x_5$
- Table

OUTPUT

- 9 constraints :



#q = 5

Results

- We implemented GENACQ and plugged it in the constraint acquisition system QUACQ for G-QUACQ version

- We Compared QUACQ to G-QUACQ on:
  - Zebra problem (5 types of 5 variables)
  - Sudoku (9 rows, 9 columns and 9 squares)
  - Latin Square (5 rows and 5 columns)
  - Radio link Frequency Assignment Problem (5 stations and 5 terminals)
  - Purdey's General Store Problem (4 families, 4 items, 4 payments)

Results

	QuAcq	g-QuAcq		
	$\#Ask$	$\#Ask$	$\#AskGen$	
Zebra	638	257	67	50%
Sudoku	8645	260	166	95%
Latin square	1129	117	60	84%
RFLAP	1653	151	37	88%
Purdey	173	82	31	34%

Strategies

- **Query Selection Heuristics**
  - Max constraints — 6 queries
  - Max variables — 5 queries
  - Min constraints — 5 queries
  - Min variables — 5 queries
  - Random

- **Cutoffs**
  - exit GENACQ before having proved the maximality
  - Cutoff on the number of consecutive negative answers

#VAR	#CST	x_2	x_5	
5	4	x_2	color	✔
25	24	x_2	X	✘
5	4	color	x_5	✔
5	10	color	color	✔
25	110	color	X	✘
25	24	X	x_5	✘
25	110	X	color	✘
25	300	X	X	✘

Results(1/3)

- G-QUACQ with heuristics and cutoff strategy on Sudoku

	cutoff	$\#Ask$	$\#AskGen$	$\#yes$	$\#no$
random			166	42	124
min_VAR	$+\infty$	260	90	21	69
min_CST			132	63	69
max_VAR			263	63	200
max_CST			247	21	226
min_VAR	3	260	75	21	54
	2		57	21	36
	1		**39**	21	18
min_CST	3	626	238	112	126
	2	679	231	132	99
	1	837	213	153	60

Results(2/3)

- G-QUACQ with random, min_VAR, and cutoff=1 on Zebra, Latin square, RLFAP, and Purdey

	$\#Ask$	$\#AskGen$	$\#yes$	$\#no$
Zebra				
Random	257	67	10	57
min_VAR		48	5	43
min_VAR +cutoff=1		**23**	5	18
Latin square				
Random	117	60	16	44
min_VAR		34	10	24
min_VAR +cutoff=1		**20**	10	10

Results(3/3)

- G-QUACQ with random, min_VAR, and cutoff=1 on Zebra, Latin square, RLFAP, and Purdey

	$\#Ask$	$\#AskGen$	$\#yes$	$\#no$
RLFAP				
Random	151	37	16	21
min_VAR		41	14	27
min_VAR +cutoff=1		**22**	14	8
Purdey				
Random	82	31	5	26
min_VAR		24	3	21
min_VAR +cutoff=1		**12**	3	9

Conclusions

- Generalization query based on types of variables
- GENACQ algorithm
- Several heuristics and strategies to select the good candidate generalization query
- Can be plugged in any active constraint acquisition system
- Results by plugging GENACQ in the QUACQ acquisition System