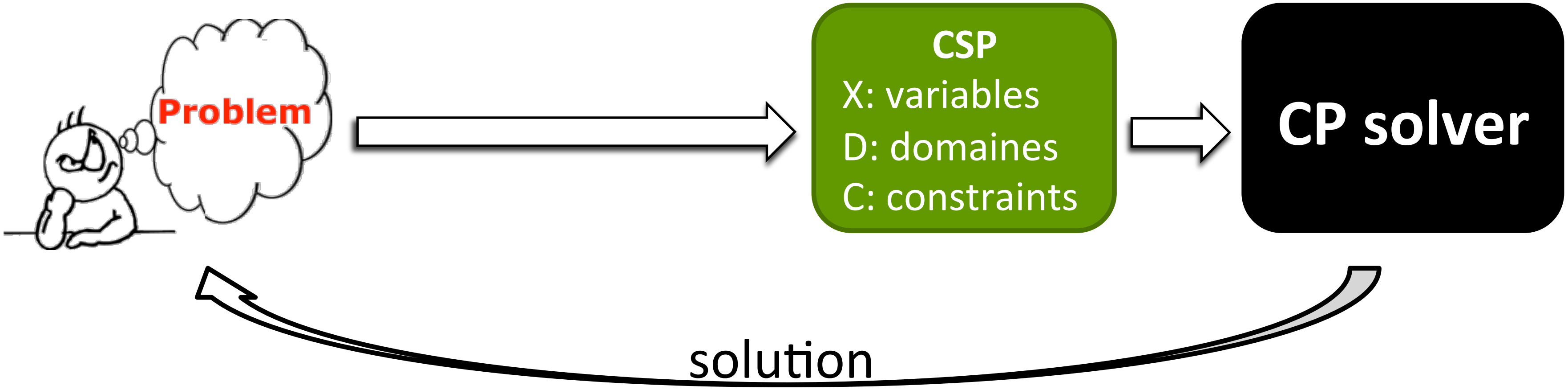
Solve a Constraint Problem Without Modeling It

Christian Bessiere, Remi Coletta, **Nadjib Lazaar**

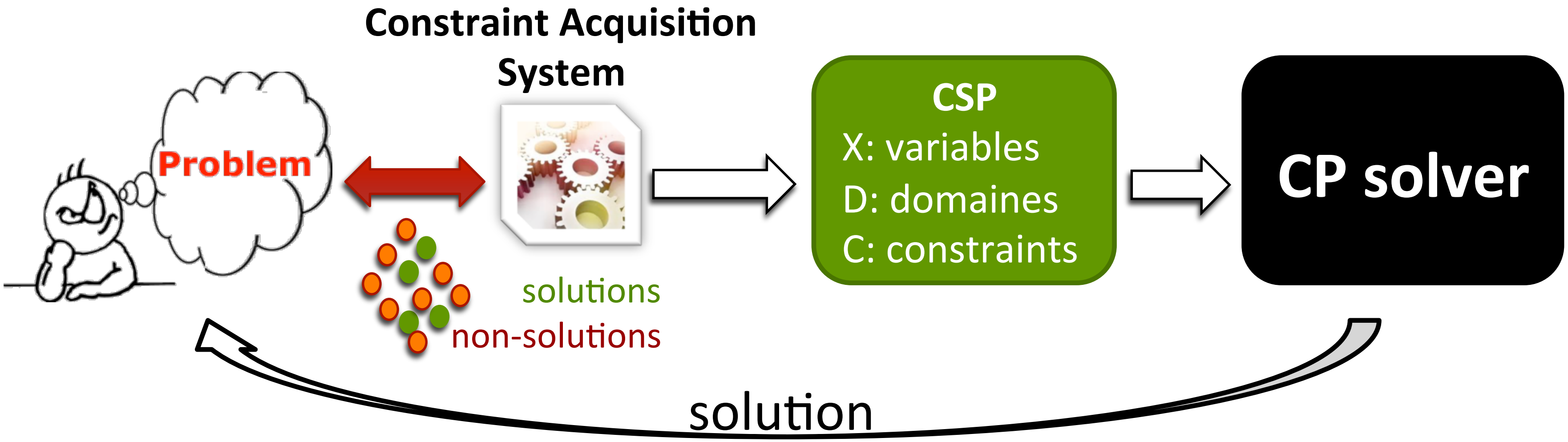
CNRS, University of Montpellier
COCONUT Team

Motivations



- Limitations: modelling constraint networks require a fair expertise

Motivations



Constraint Acquisition System

Problem

solutions

non-solutions

CSP

X: variables

D: domaines

C: constraints

CP solver

solution

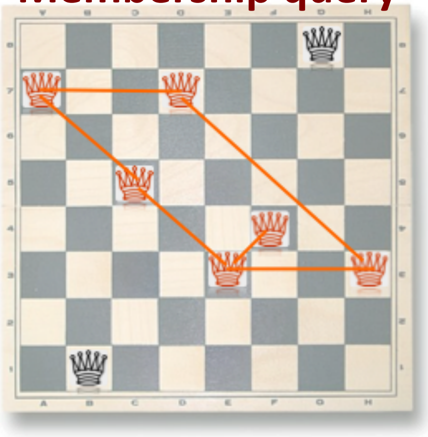
- Limitations: modelling constraint networks require a fair expertise

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

Membership query

ask([2,8,4,2,6,5,1,6])=No



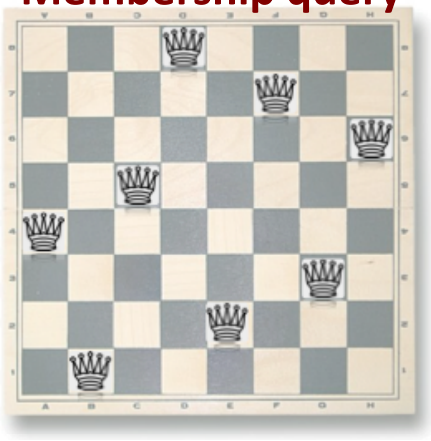
✗

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

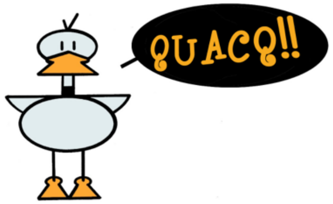
Membership query

ask([5,8,4,1,7,2,6,3])=Yes



✓

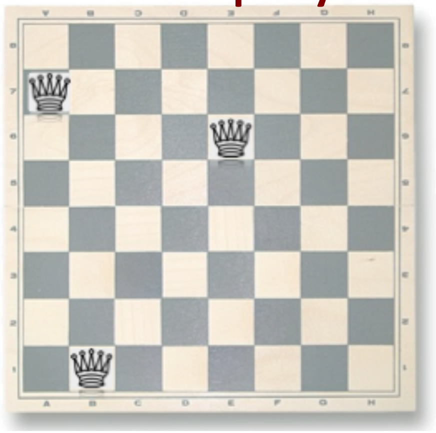
QUACQ: Quick Acquisition [Bessiere et al. 13]



- Active learning approach
- Based on partial queries to elucidate the scope of the constraint to learn

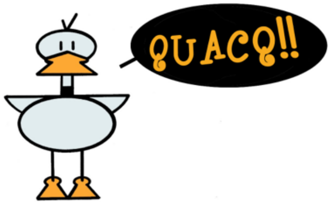
Partial query

ask([5,8,4,2,3,_,_,_])=Yes



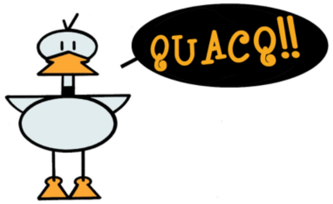
✓

QUACQ: Quick Acquisition [Bessiere et al. 13]



- Active learning approach
- Based on partial queries to elucidate the scope of the constraint to learn
- QUACQ does not require complete positive examples
  → we can use it to ***solve*** an instance

QUACQ: Quick Acquisition [Bessiere et al. 13]



- Active learning approach
- Based on partial queries to elucidate the scope of the constraint to learn
- QUACQ does not require complete positive examples
  → we can use it to ***solve*** an instance

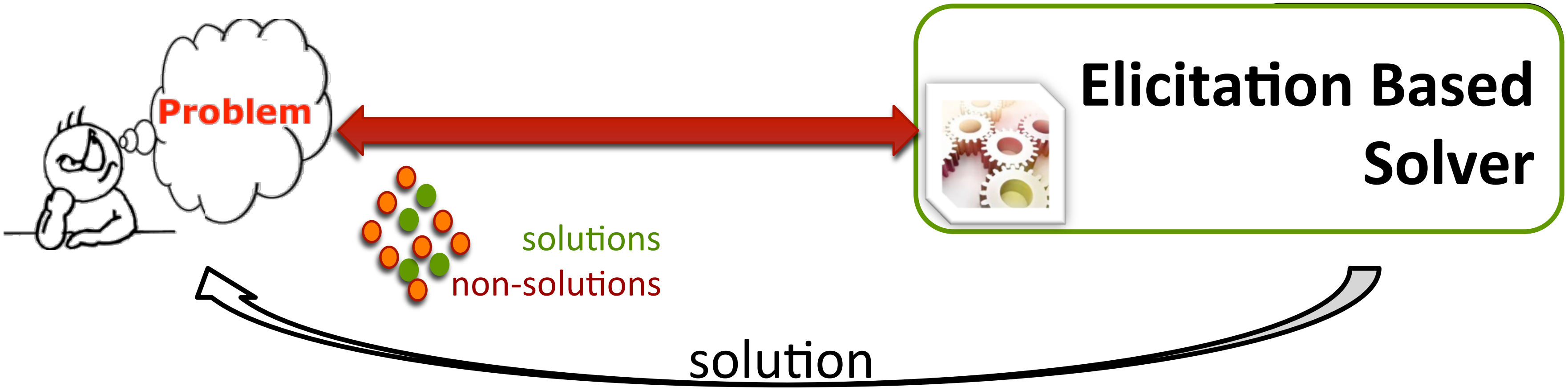
Limitation:

- QUACQ promotes **learning** and it can find a solution by chance!

Question:

- In a constraint acquisition context, can we promote **solving**?

Motivations



Problem

solutions

non-solutions

Elicitation Based Solver

solution

- Limitations: modelling constraint networks require a fair expertise
- Question: Can we solve a problem without modelling it?

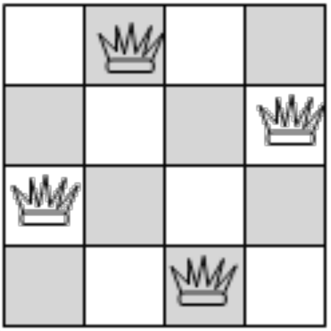
Ask&Solve

- Objective:
  - **Solving** a problem without having a constraint network describing it
  - Find the best tradeoff between **learning** and **solving** to converge as soon as possible on a solution

- How:
  - Asking (partial) queries
  - **Extend** a scope on which we know at least one assignment accepted by the target network $\mathbf{C}_T$
  - **Learn** a culprit constraint at each negative example, to prune the search space (QUACQ-like process)

Ask&Solve

- Example (4-queens)



learn	e=	1	1			✗	$Cst = \{q1 \neq q2\}$
learn		1	2			✗	$Cst = Cst \cup \{q2 \neq q1+1\}$
extend		1	3			✓	
learn		1	3	1		✗	$Cst = Cst \cup \{q1 \neq q3\}$
			⋮				
extend		2	4	1		✓	
learn		2	4	1	1	✗	$Cst = Cst \cup \{q3 \neq q4\}$
Solution!		2	4	1	3	✓	

Experiments

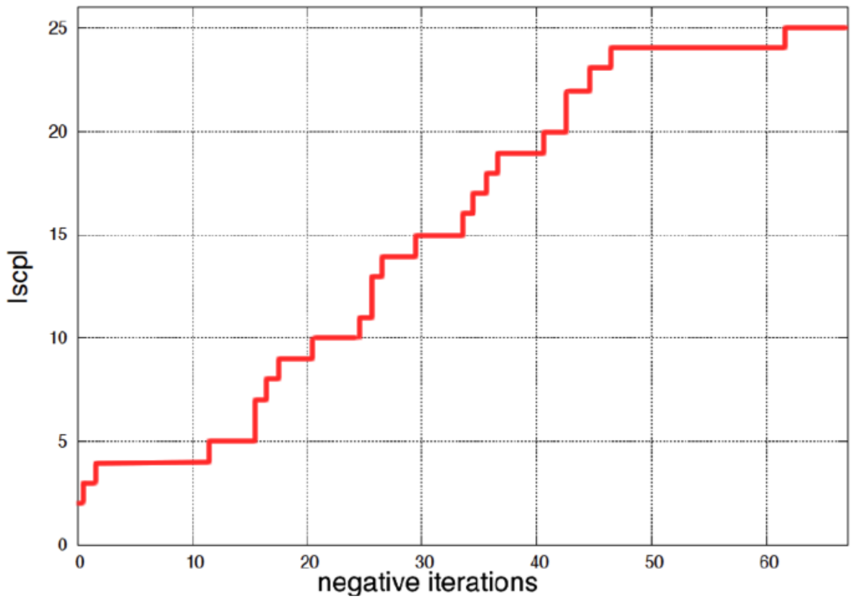
- A comparative study with
  - Baseline 1: QUACQ&Solve
  - Baseline 2: Branch&Learn [Bessiere et al. 12]
    - is a backtrack search based on elicitation (asking queries at each node)
    - Use of CONACQ at each node
  - Baseline 3: Backtrack-E
    - If the query is classified as positive we reduce the version space
    - Negative, we learn a constraint using the QUACQ principle

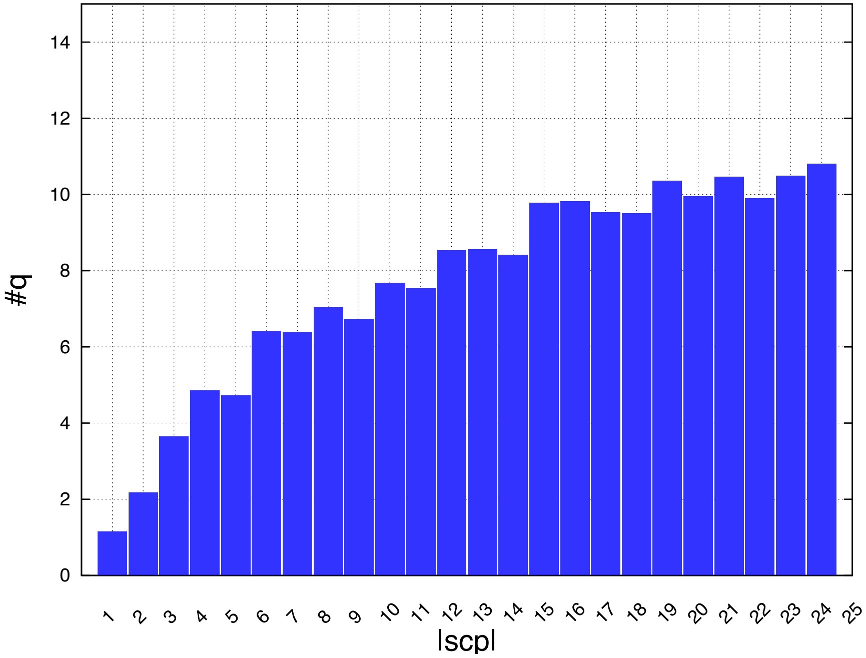
Experiments

		$\#Csts$	$\#queries$	$time\backslash queries$
Golomb	QuAcq&Solve	111	548	0.21
	Backtrack-E	46	432	0.16
	Branch&Learn	–	389	76.01
	Ask&Solve	21	**179**	0.35
Zebra	QuAcq&Solve	58	623	0.02
	Backtrack-E	51	528	0.06
	Branch&Learn	–	—	—
	Ask&Solve	60	**509**	0.02
Purdey	QuAcq&Solve	18	157	0.01
	Backtrack-E	15	119	0.01
	Branch&Learn	–	109	0.61
	Ask&Solve	14	**103**	0.01

Ask&Solve behavior

- Zebra problem





- How can we reduce the Area Under the Curve (#queries)?

→ Strategies (restart policies / variable ordering heuristics)

Restart policies

- FC-restart (fixed cutoff)
- Geometric-restart [Walsh. 99]
- Luby-restart [Luby et al. 93]

Variable ordering heuristics

- Random: At each restart event, we reorder the variables randomly
- Lexicographic (lex):

$$x_1, x_2, x_3 \xrightarrow{\text{restart}} x_1, x_2, x_3, x_4 \ldots$$

- Reverse-lex (r-lex):

$$x_1, x_2 \xrightarrow{\text{restart}} x_2, x_1, x_3, x_4 \xrightarrow{\text{restart}} x_4, x_3, x_1 \ldots$$

- Continuous-lex (c-lex):

$$x_1, x_2, x_3 \xrightarrow{\text{restart}} x_3, x_4 \xrightarrow{\text{restart}} x_4, x_5, x_6 \ldots$$

Results (with strategies)

	RESTART	VAR-ORDER	$\#Csts$	$\#queries$	$time\backslash queries$
Golomb	none	LEX	21	179	0.35
	FC	RANDOM	48	435	0.24
		LEX	21	174	0.34
		R-LEX	30	232	0.30
		C-LEX	28	203	0.35
	Geometric	RANDOM	56	527	0.27
		LEX	21	202	0.33
		R-LEX	21	166	0.28
		C-LEX	21	162	0.31
	Luby	RANDOM	45	402	0.31
		LEX	21	161	0.34
		R-LEX	21	160	0.33
		C-LEX	11	**158**	0.32

Results (with strategies)

	RESTART	VAR-ORDER	$\#Csts$	$\#queries$	$time\backslash queries$
Zebra	none	LEX	60	509	0.02
	FC	RANDOM	57	560	0.05
		LEX	63	558	0.02
		R-LEX	53	452	0.05
		C-LEX	59	459	0.03
	Geometric	RANDOM	59	503	0.02
		LEX	60	482	0.05
		R-LEX	48	**346**	0.03
		C-LEX	59	381	0.04
	Luby	RANDOM	57	484	0.05
		LEX	60	537	0.03
		R-LEX	41	356	0.03
		C-LEX	57	465	0.02

Results (with strategies)

	RESTART	VAR-ORDER	$\#Csts$	$\#queries$	$time\backslash queries$
Purdey	none	LEX	14	103	0.01
	FC	RANDOM	16	106	0.01
		LEX	13	108	0.01
		R-LEX	11	88	0.02
		C-LEX	12	82	0.01
	Geometric	RANDOM	16	99	0.02
		LEX	12	77	0.01
		R-LEX	8	**37**	0.02
		C-LEX	15	64	0.01
	Luby	RANDOM	16	123	0.01
		LEX	12	86	0.01
		R-LEX	9	62	0.02
		C-LEX	11	83	0.01

Conclusion

- QUACQ can be used as a solver but it promotes **learning**
- We present Ask&Solve, an elicitation based solver that promotes **solving**
  - Solving without the need of a constraint network
- Ask&Solve can be boosted using restart policies and variable orderings

➔ Decrease even more the number of queries by plugging other techniques (ModelSeeker, Complexe queries…)