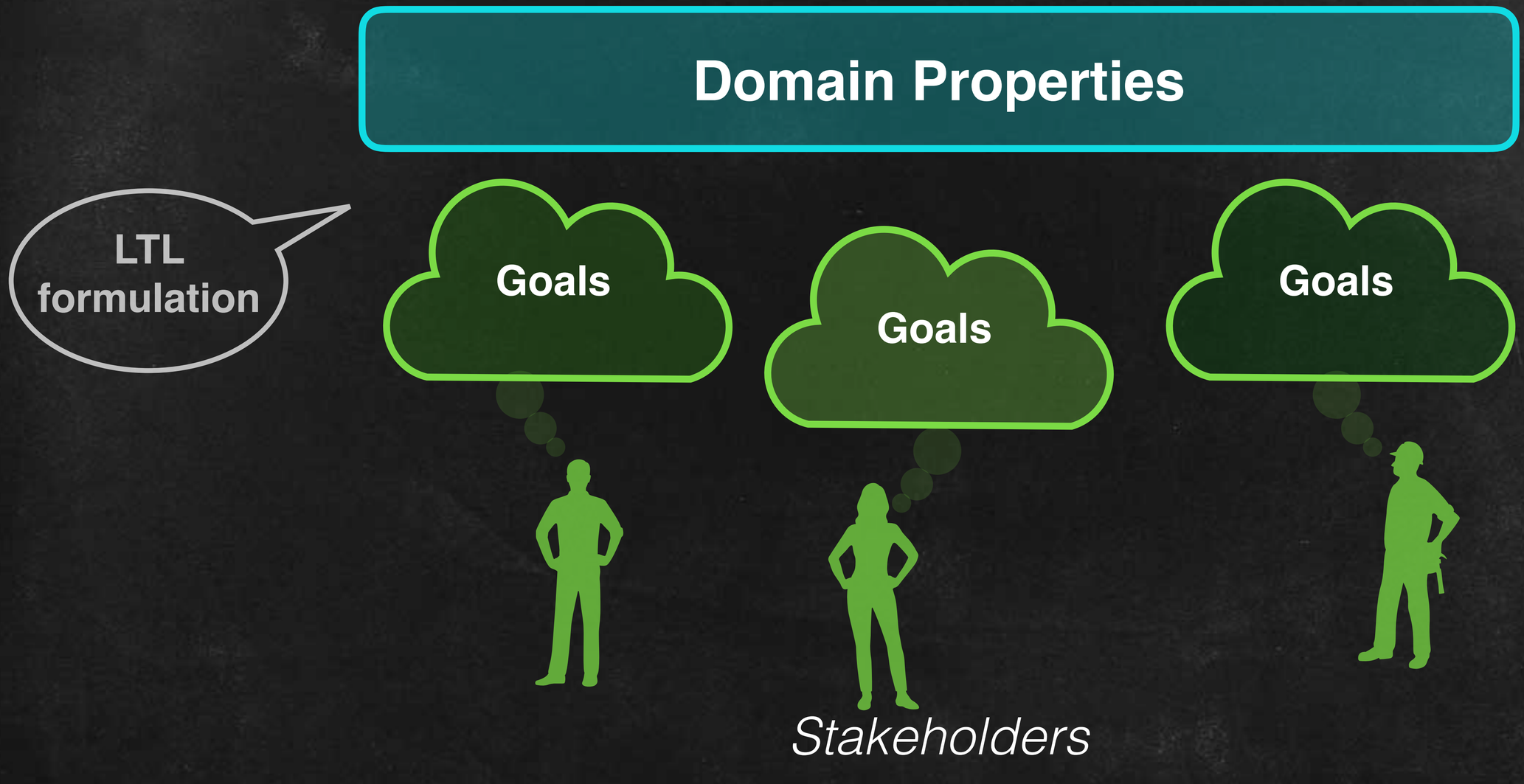
A Genetic Algorithm for Goal-Conflict Identification

Facundo Molina

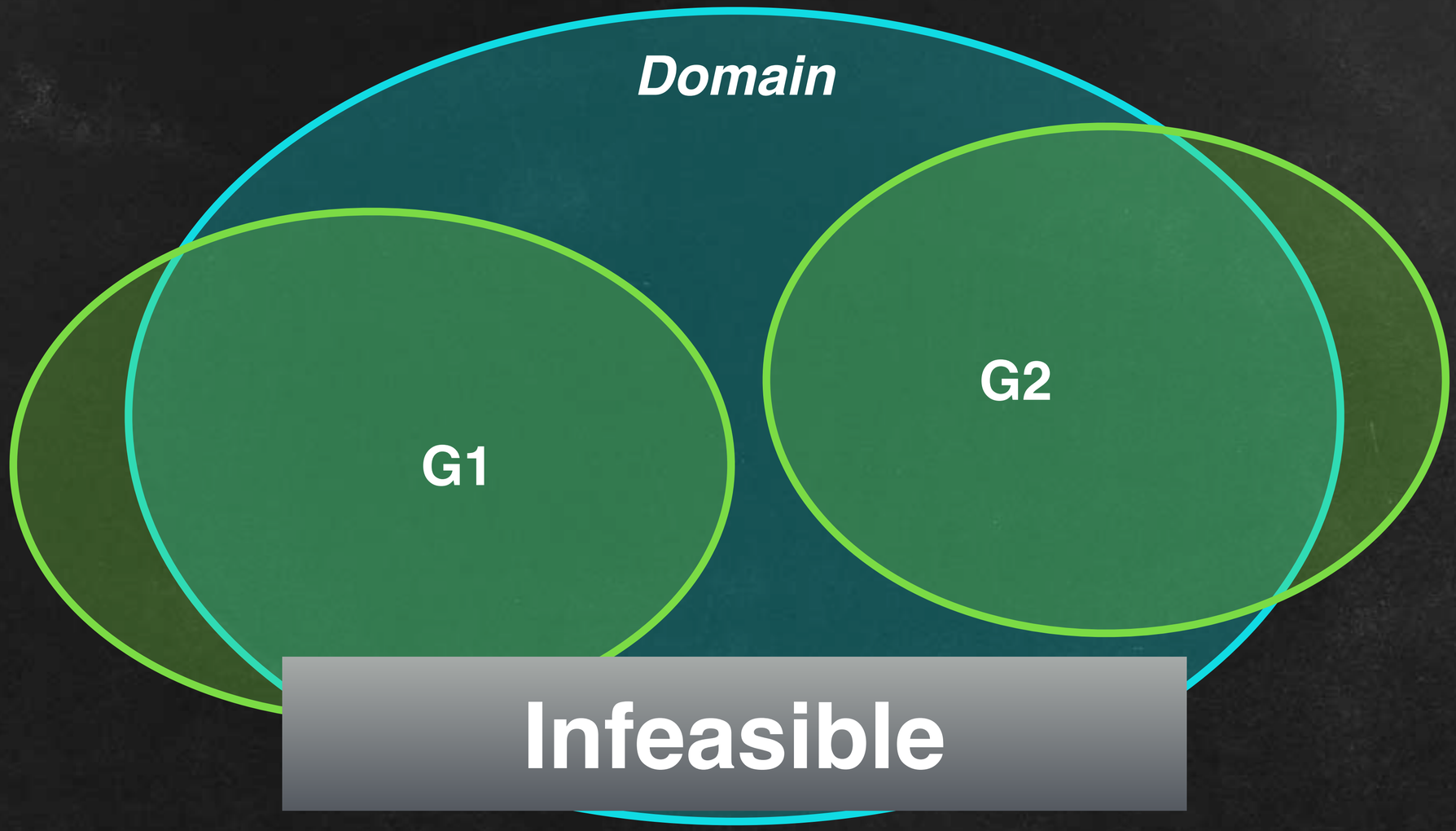
ASE 2018

in collaboration with
Renzo Degiovanni, Germán Regis and Nazareno Aguirre

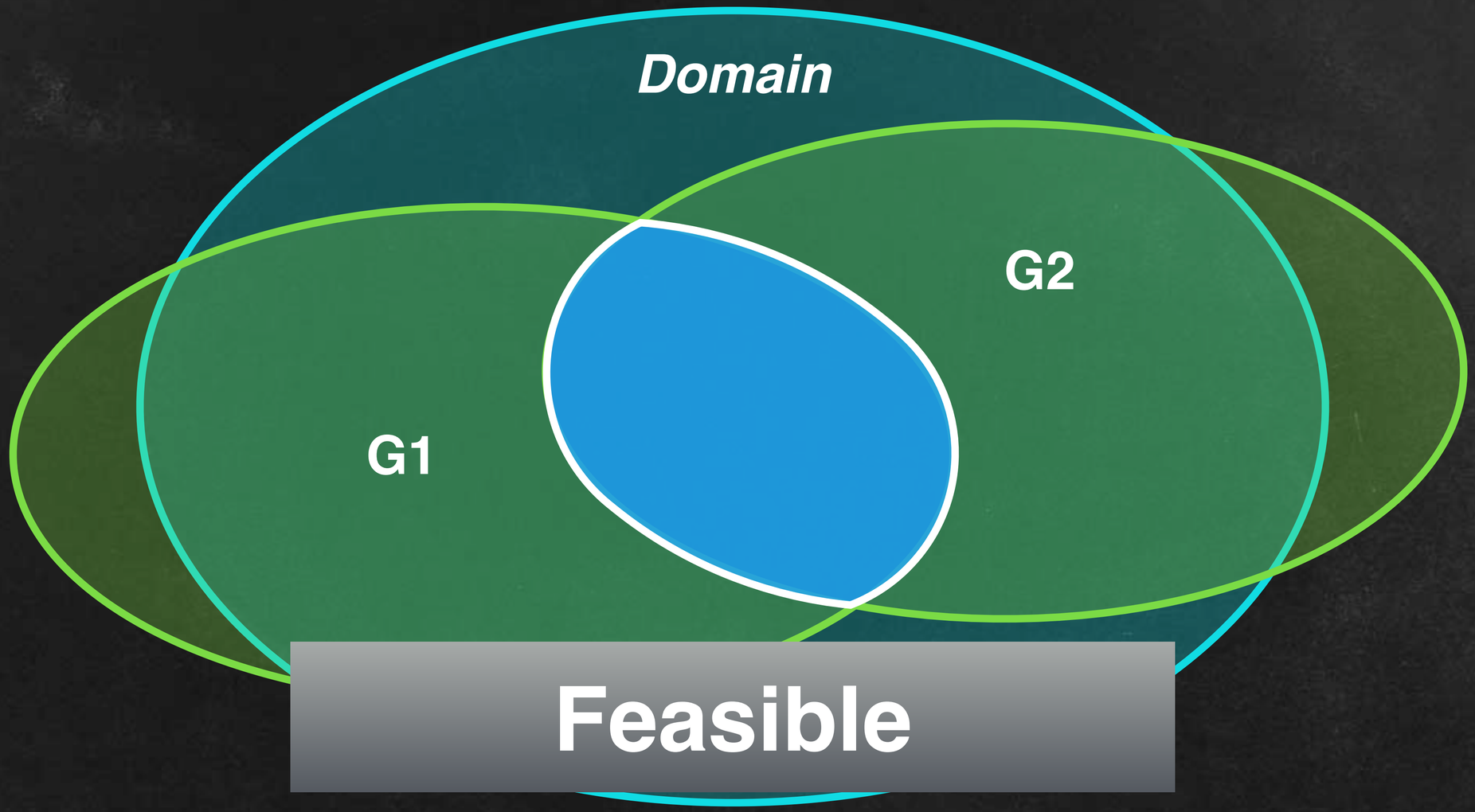
Early phase in the RE process



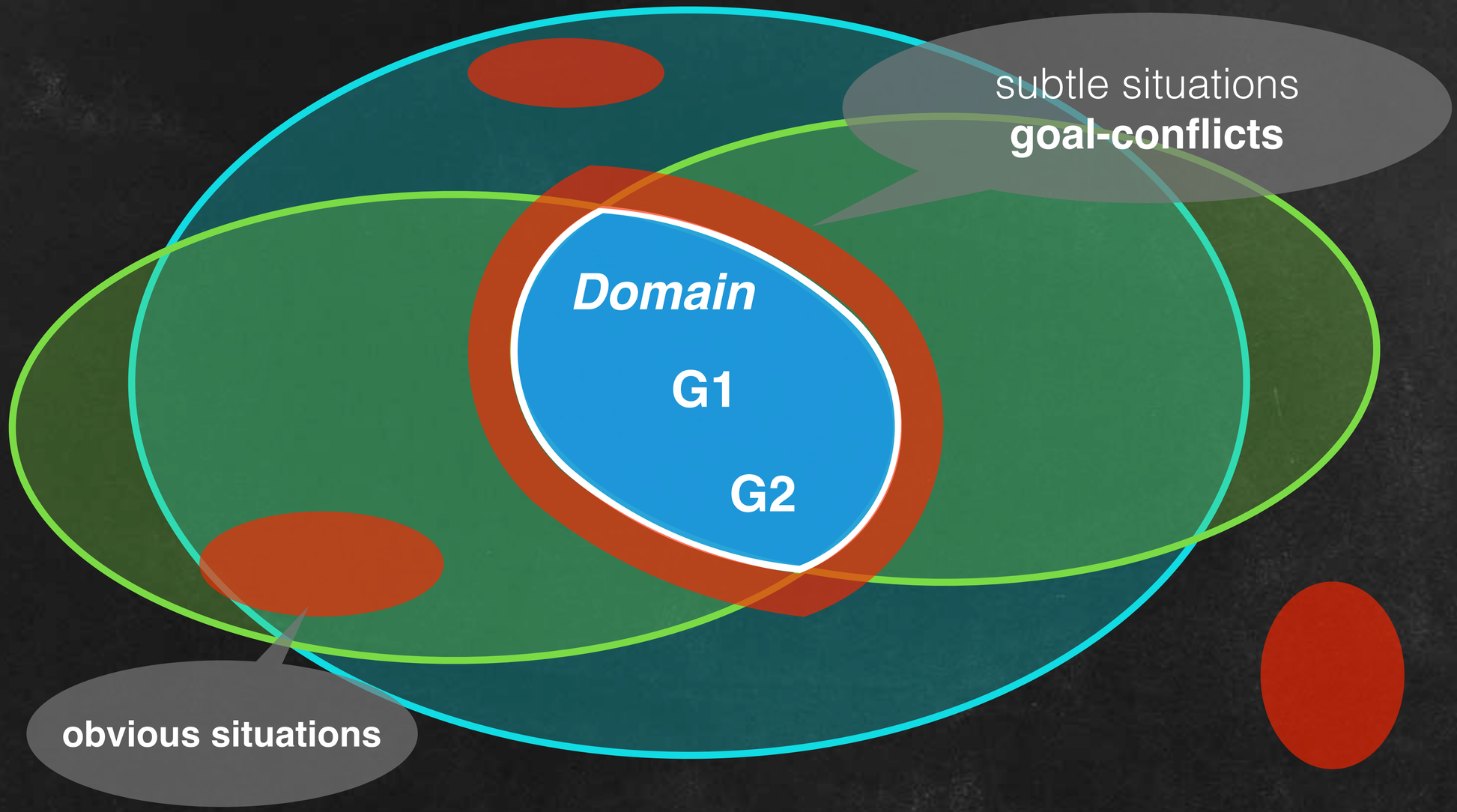
Feasibility



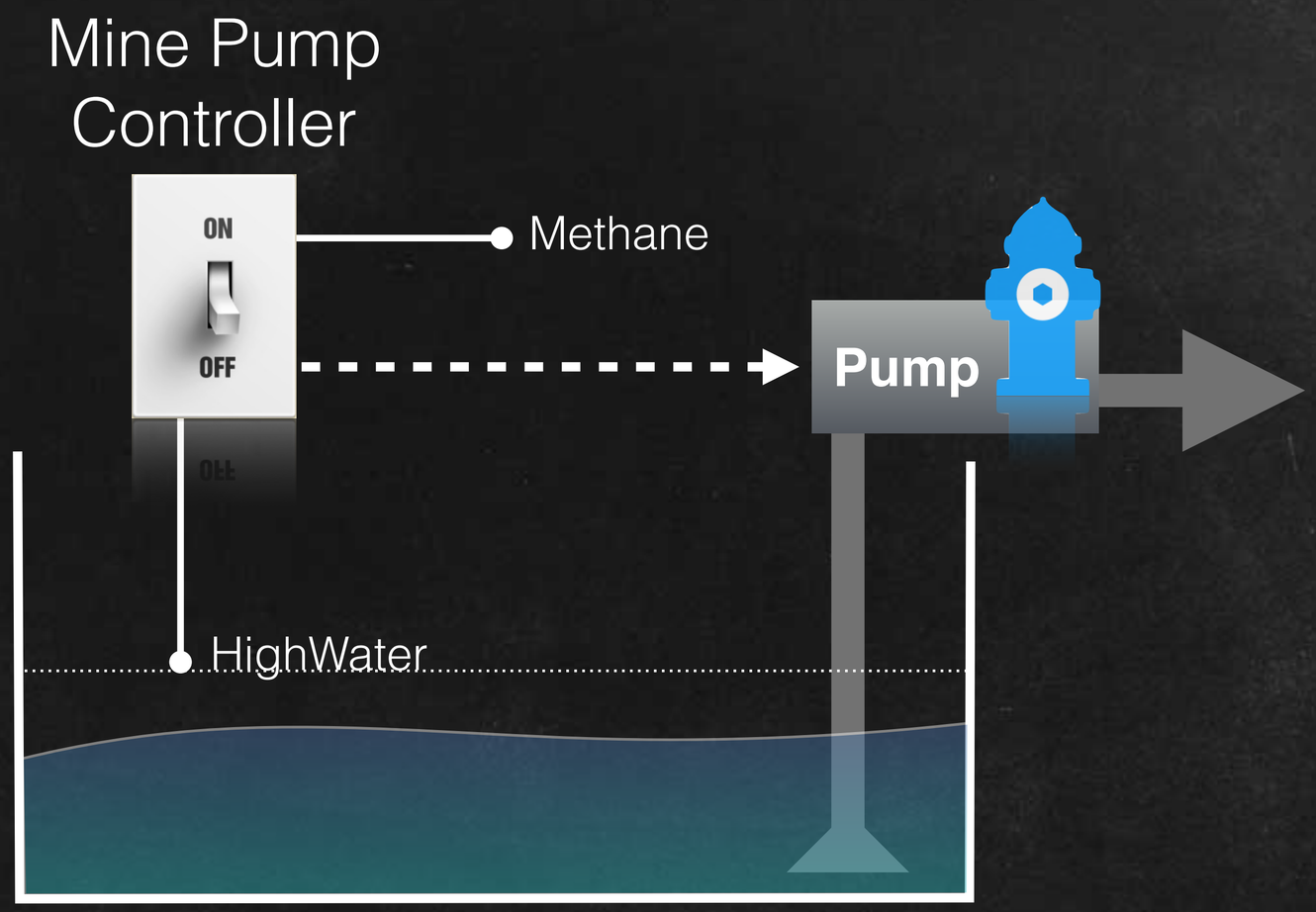
Feasibility



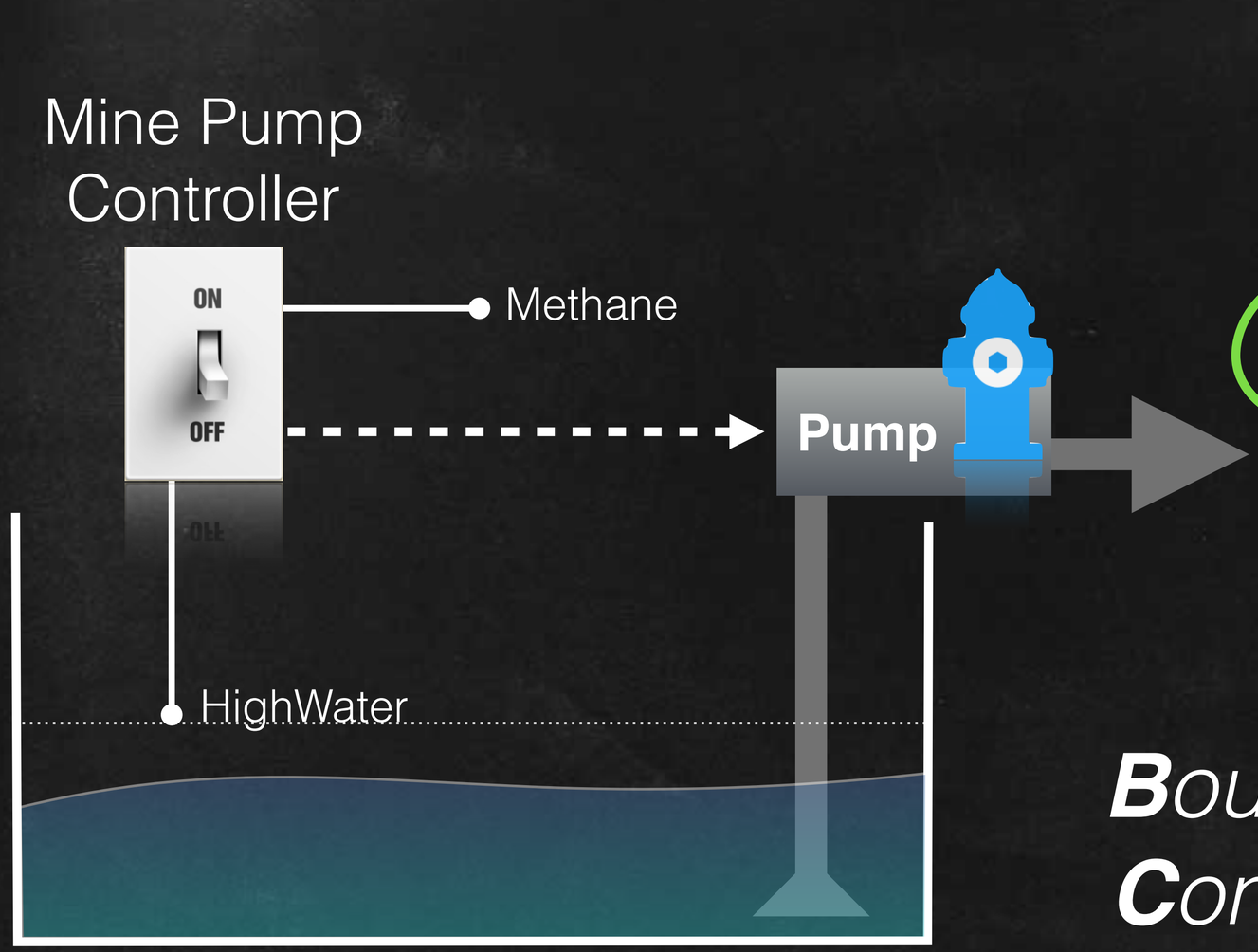
In which situations the goals can’t be fulfilled?



Mine Pump Controller



Mine Pump Controller



If **PumpOn**, then **not HighWater** in at most two minutes

If **Methane,** then **not PumpOn**

If **HighWater,** then **PumpOn**

***B**oundary*
***C**ondition*

Methane and HighWater

Divergences

the goals are **divergent** w.r.t. the domain iff there exists a ***boundary condition BC*** such that:

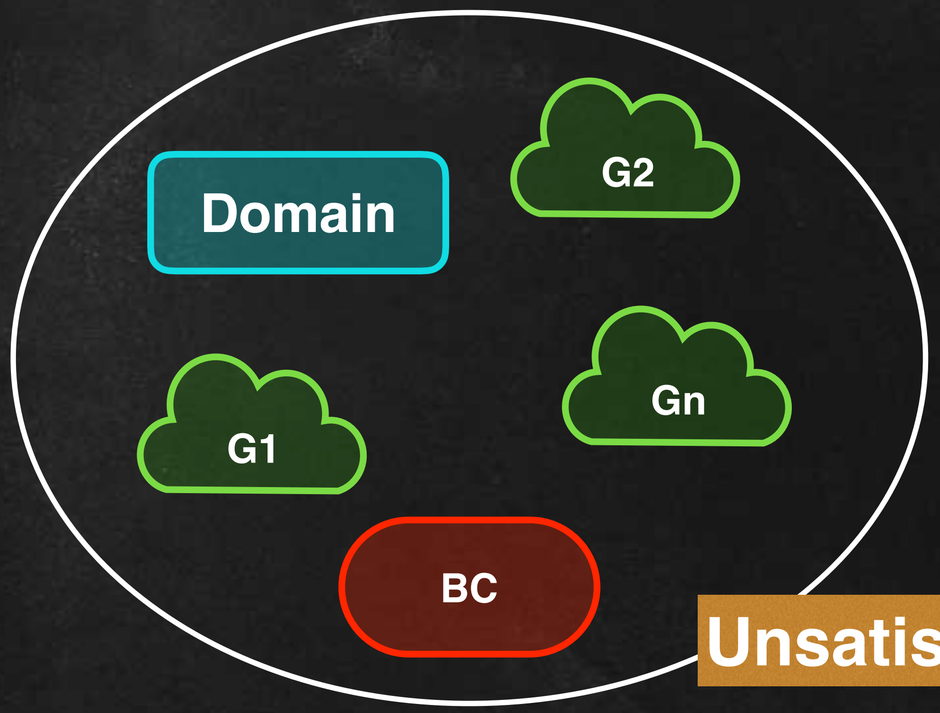
(1) logical inconsistency

(2) minimality

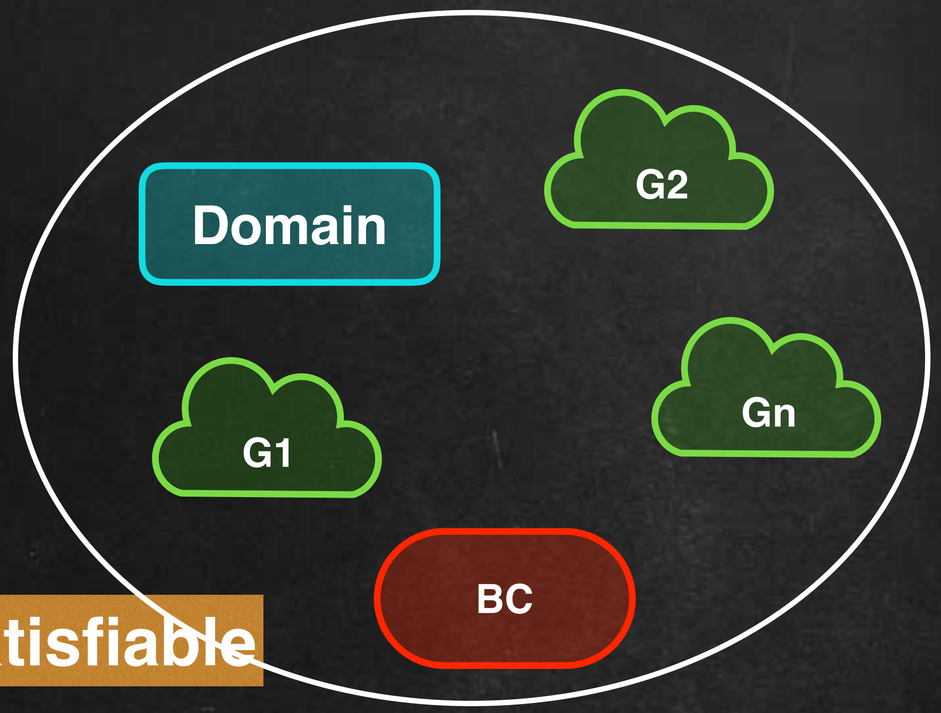
(3) non-triviality

Divergencies

(1) logical inconsistency



(2) minimality

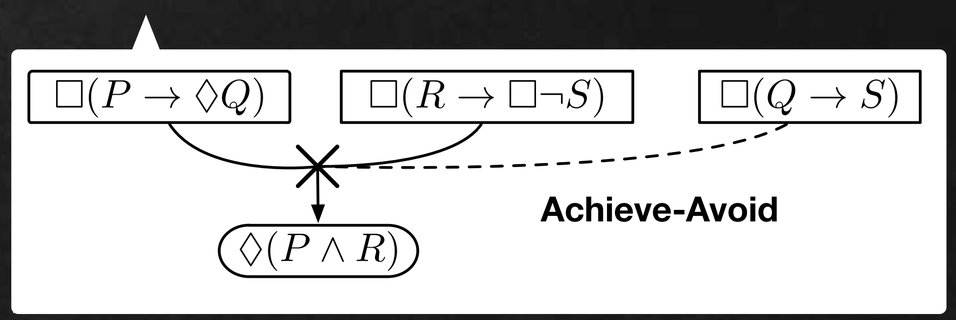


(3) non-triviality $BC \neq \neg(G_1 \wedge \ldots \wedge G_n)$

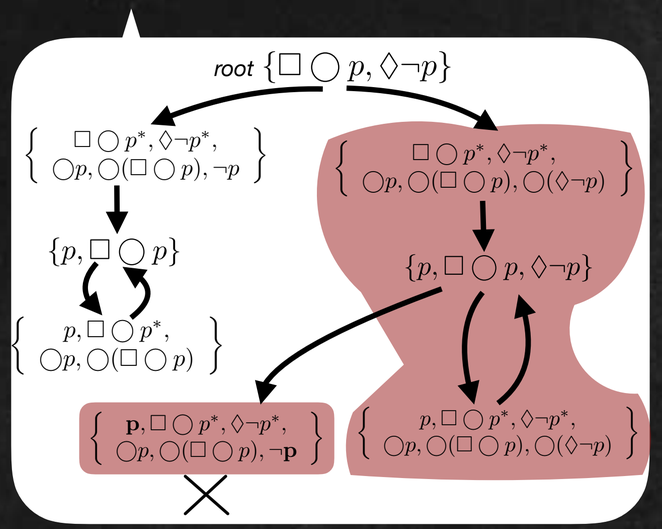
State of the Art

- Automatically identifying boundary conditions

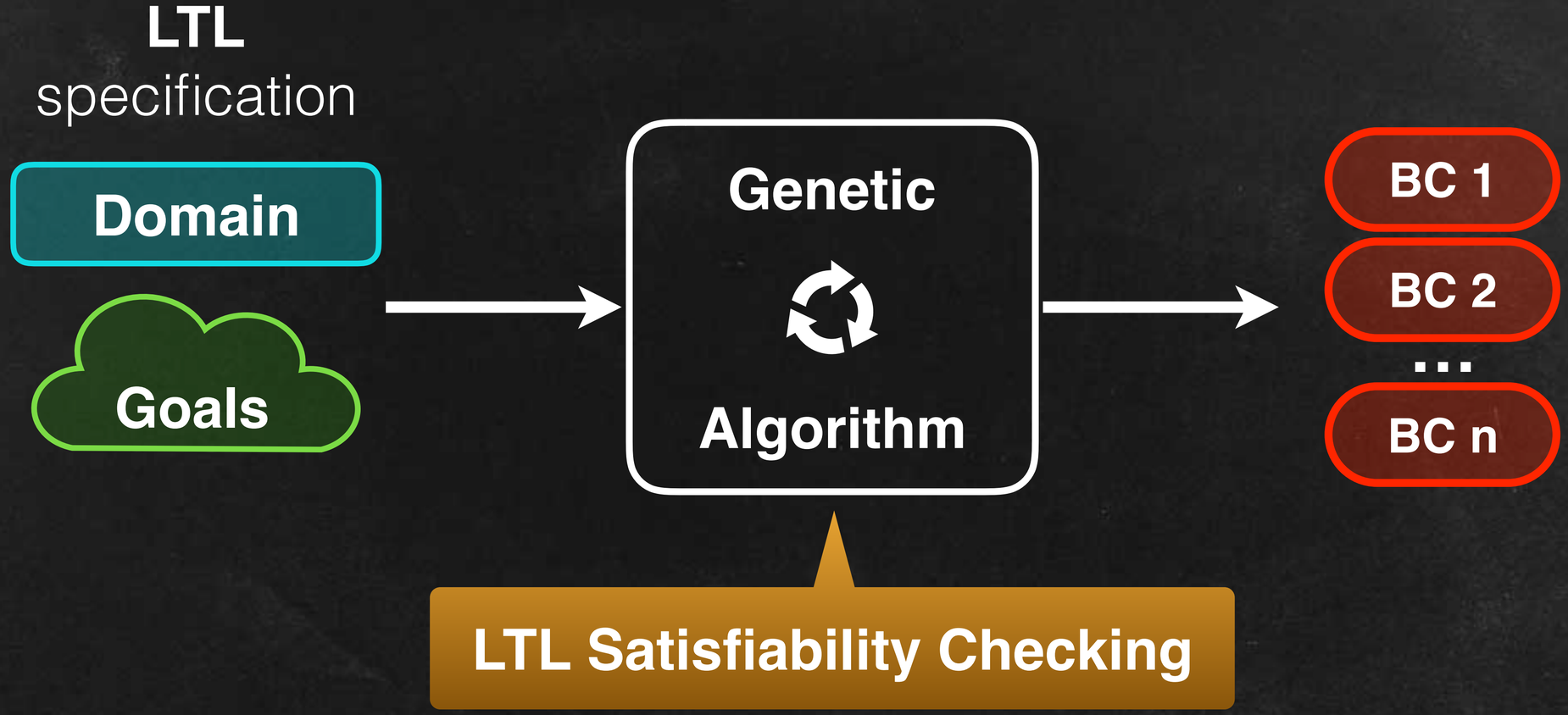
Pattern based technique **[TSE'98]**.
— restricted to captured patterns.



Tableaux based technique **[ASE'16]**
— very expensive logical manipulation of the tableau structure.



Our Proposal



Genetic Algorithm

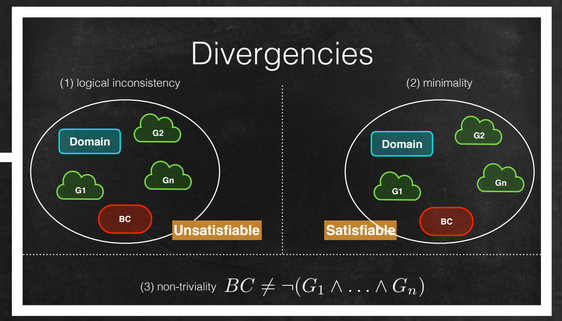
Search Space

$\Box(p \rightarrow \neg q)$ $\Diamond(r \wedge s)$

$(p \vee q)\ \mathcal{U}\ \neg r$

Initial Population

Evaluation



LTL SAT checks

Stop Criterion?

Yes

Solutions

No

Selection

Genetic Operators

Mutation

Crossover

Syntactical alterations

$\Box(p \rightarrow \neg q)$

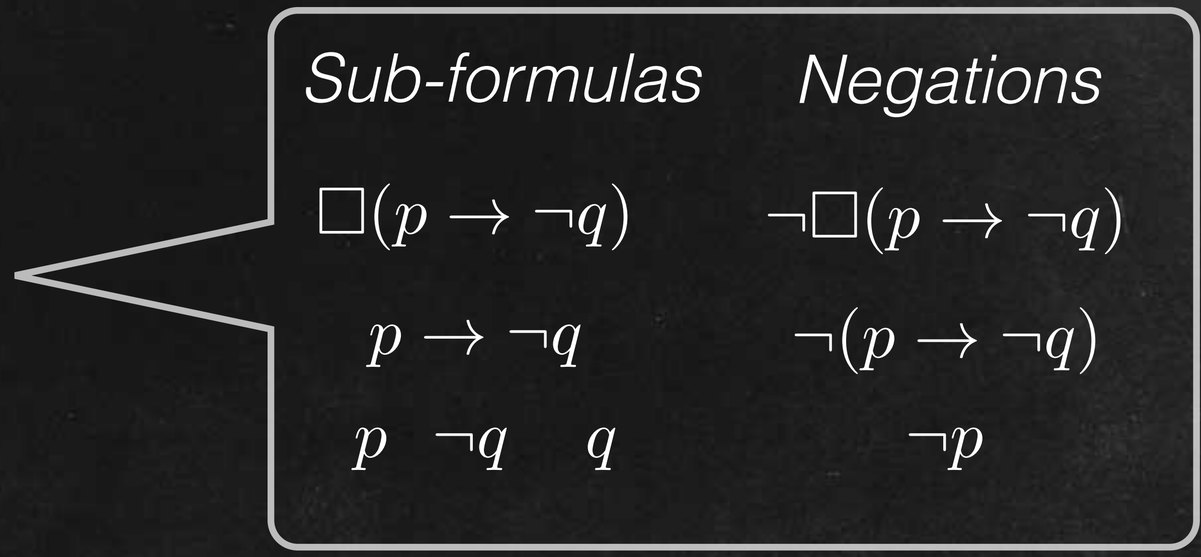
$\Box(p \wedge \neg q)$

Initial Population

- All **sub-formulas**, and their **negations**, computed from the domain properties and the goals.

Goal

$\Box(p \rightarrow \neg q)$

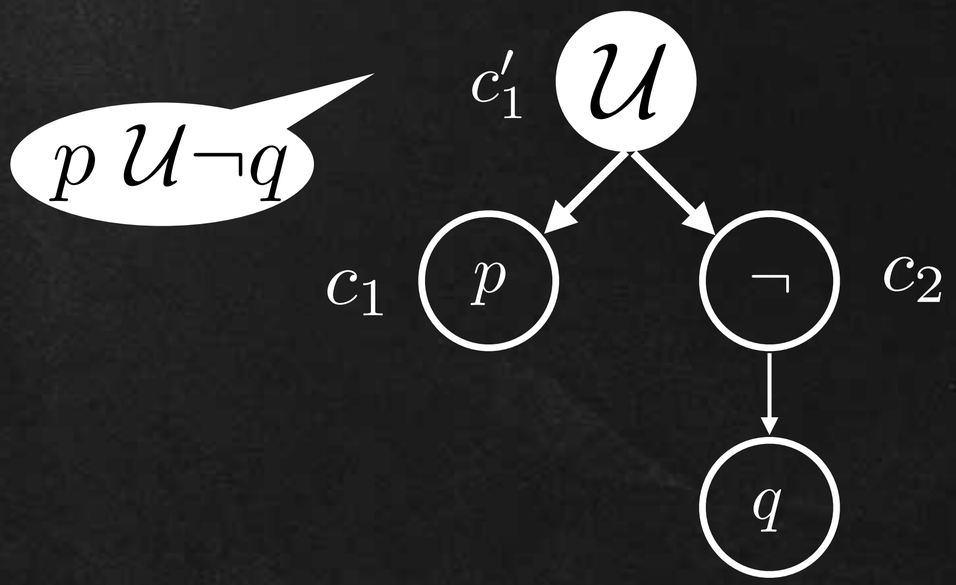


initial population

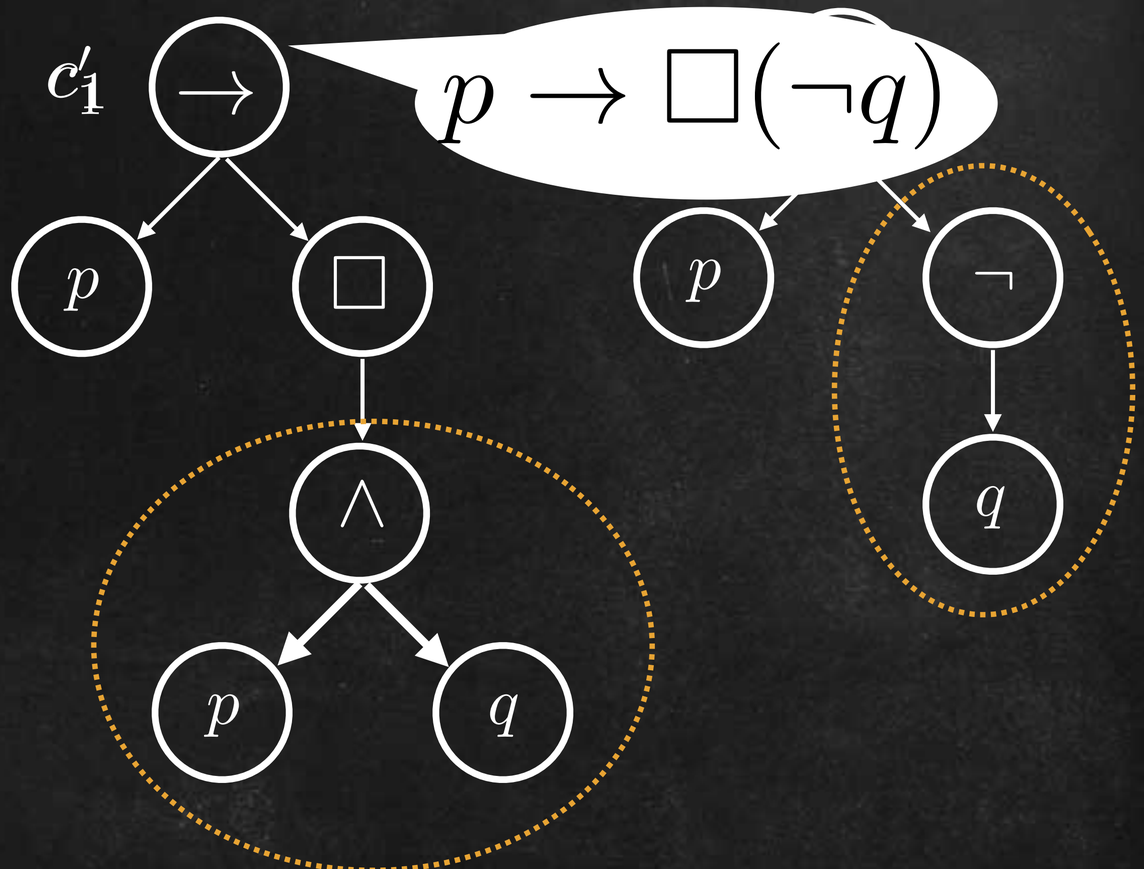
Genetic Operators

Crossover

binary combination



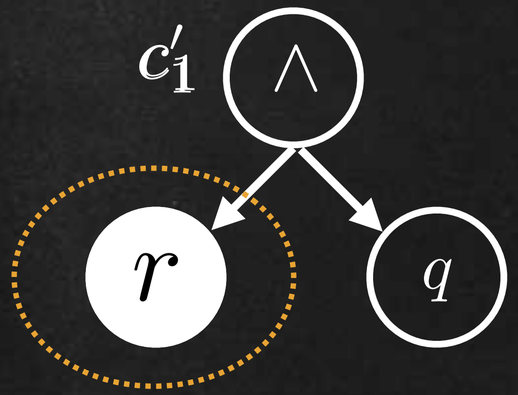
sub formula replacement



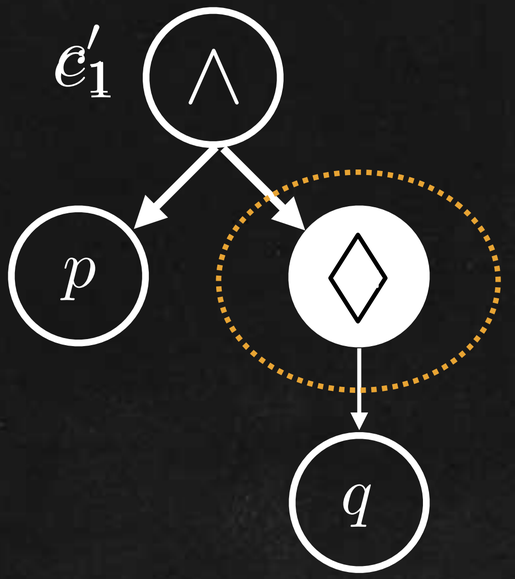
Genetic Operators

Mutation

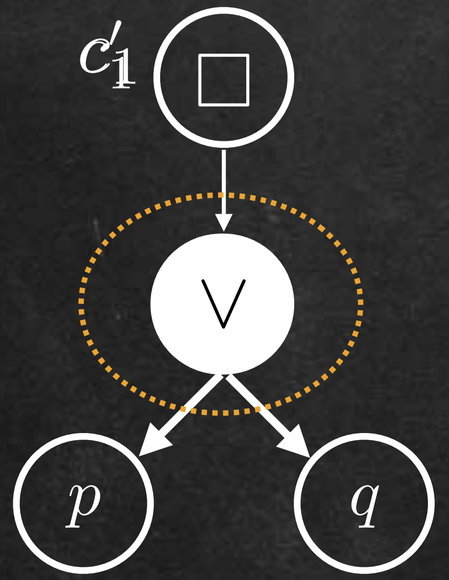
atomic replacement



unary op. replacement



binary op. replacement



Fitness Function

Let φ_c be a **candidate** **boundary condition**

$$f(\varphi_c) = li(\varphi_c) + \sum_{i=1}^{|G|} min(\varphi_c, G_i) + nt(\varphi_c) + \frac{1}{\#\varphi_c}$$

logical inconsistency | minimality | non-triviality | formula size penalty

$$li(\varphi_c) = \begin{cases} 1 & \text{if} \\ & \\ 0 & \text{(} \end{cases} \qquad min(\varphi_c, G_i) = \begin{cases} \frac{1}{|G|} \\ 0 \end{cases} \qquad nt(\varphi_c) = \begin{cases} 0.5 & \text{if } \varphi_c \neq \neg(G_1 \wedge \ldots \wedge G_n) \\ 0 & \text{otherwise} \end{cases}$$

the shorter, the better

Evaluation

RQ1 How effective and efficient is our approach to identify boundary conditions in requirement specifications?

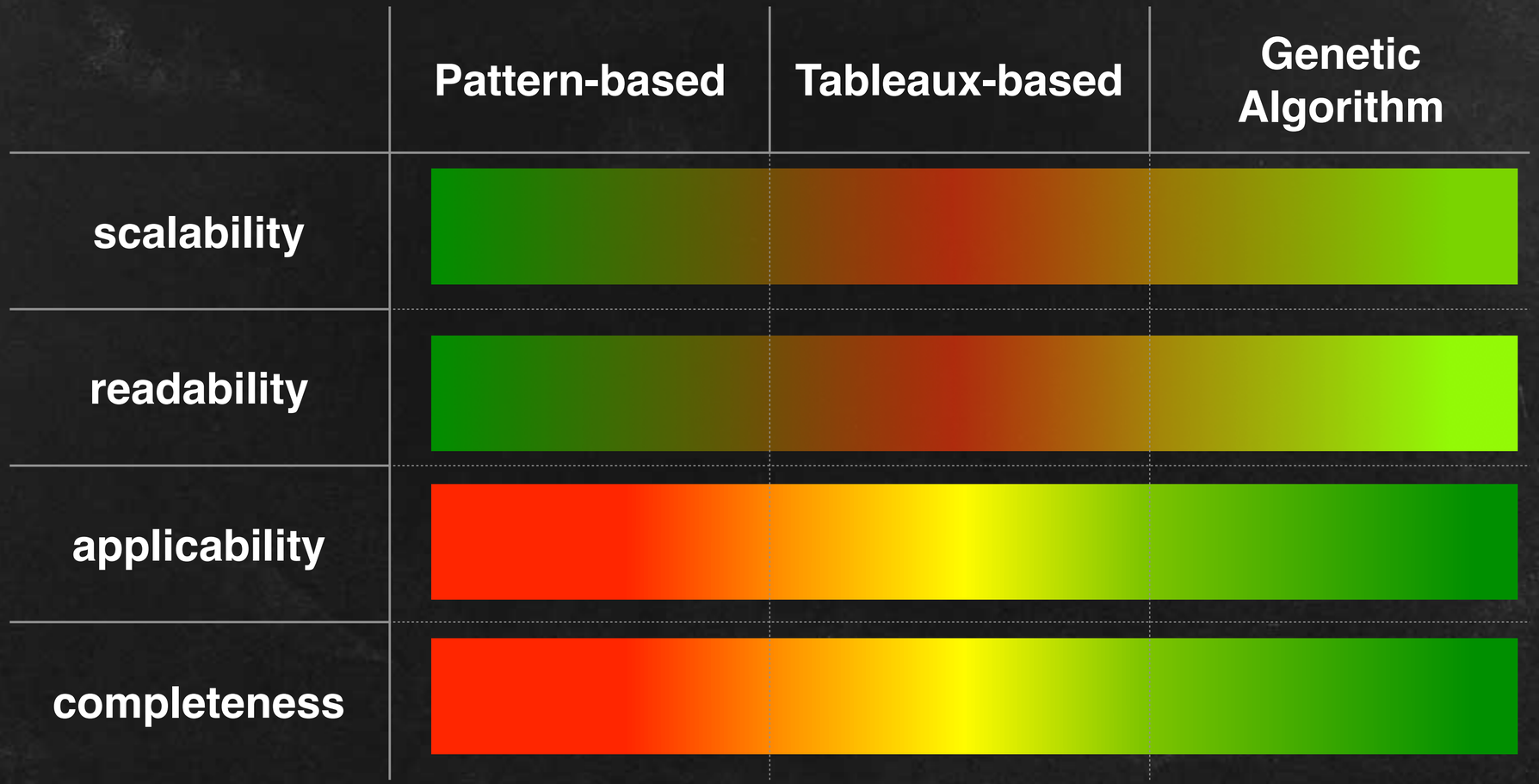
RQ2 Is our approach able to identify boundary conditions that cannot be derived by related techniques?

http://dc.exa.unrc.edu.ar/staff/rdegiovanni/ase2018.html
The tool: **JGAP**, **LTL2Buchi**, and **aalta** LTL solver.

Evaluation

Case Study	Pattern-based	Tableaux-based	Genetic Algorithm
Achieve-Avoid	1	4	21
Retraction 1	1	1	27
Retraction 2	1	1	22
RailRoadCrossingSystem	-	1	16
MinePump	-	2	18
ATM	-	4	10
Elevator	-	1	7
TCP protocol	-	2	8
Telephone	-	1	24
London Ambulance System	-	1	84
Simple Arbiter	-	TO	15
Prioritized Protocol	-	TO	13
Round Robin Arbiter	-	TO	37
Load Balancer	-	TO	3
Lift Controller	-	TO	3
AMBA	-	TO	2

Summary


Pattern-based
Tableaux-based
Genetic Algorithm
scalability
readability
applicability
completeness

Applicability and Usability

Control Synthesis Problem

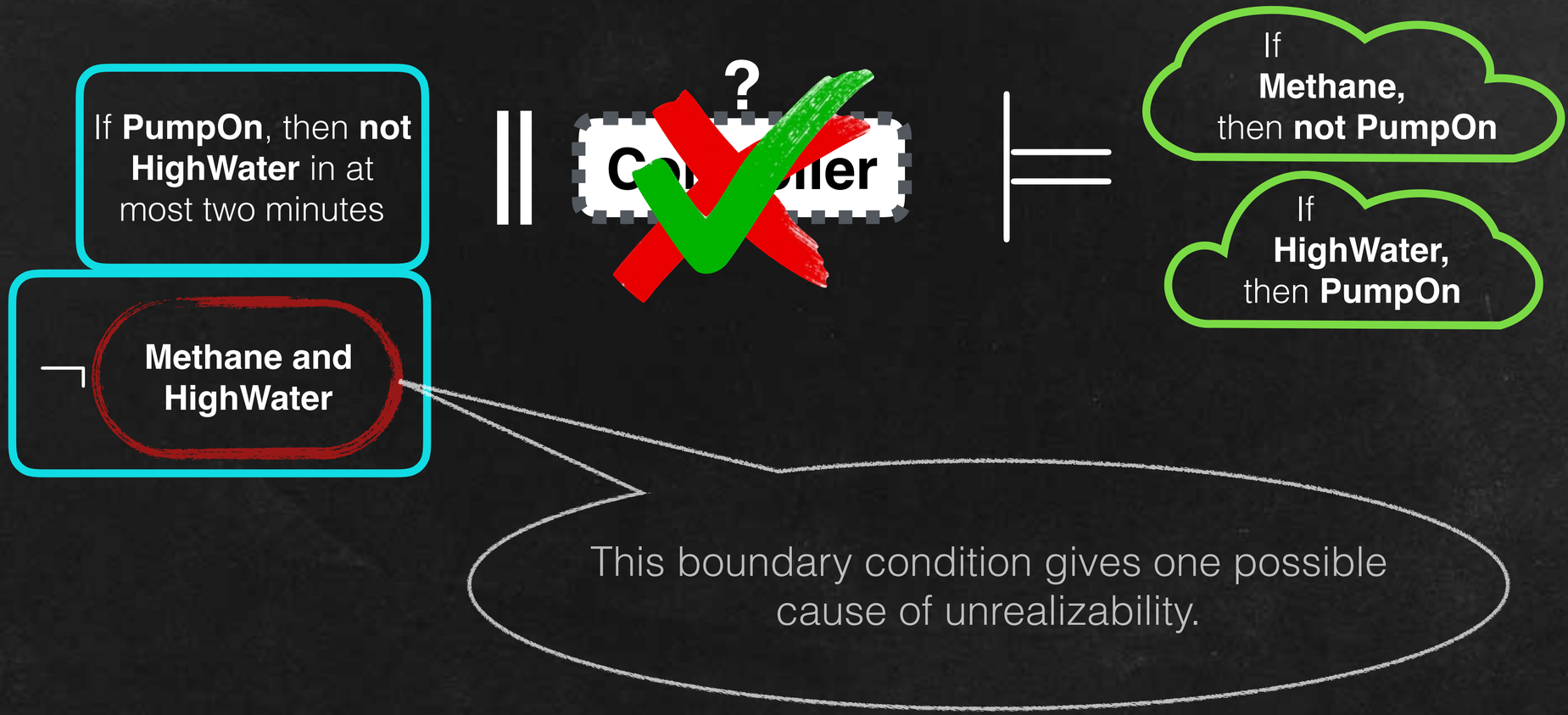
✔ realizable

x unrealizable

Can *boundary conditions* **explain** why the specifications are **unrealizable**?

Applicability and Usability

Mine Pump Controller



Remarks

- Novel application of genetic algorithms in the context of software engineering
- More general and scalable automated technique for boundary condition computation
- Enables the application of boundary conditions for requirements engineering problems with increased demands of scalability

Questions?

Thanks

Evaluation (with spec sizes)

Case Study	Spec size	Pattern-based	Tableaux-based	Genetic Algorithm
Achieve-Avoid	3	1	4	21
Retraction 1	2	1	1	27
Retraction 2	2	1	1	22
RailRoadCrossingSystem	4	-	1	16
MinePump	3	-	2	18
ATM	3	-	4	10
Elevator	2	-	1	7
TCP protocol	2	-	2	8
Telephone	5	-	1	24
London Ambulance System	5	-	1	84
Simple Arbiter	7	-	TO	15
Prioritized Protocol	7	-	TO	13
Round Robin Arbiter	9	-	TO	37
Load Balancer	11	-	TO	3
Lift Controller	21	-	TO	3
AMBA	27	-	TO	2

Time Comparison (sec.)

Case Study	Pattern-based	Tableaux-based	Genetic Algorithm
Achieve-Avoid	0	2	5
Retraction 1	0	0	17
Retraction 2	0	0	16
RailRoadCrossingSystem	-	1	17
MinePump	-	9	7
ATM	-	10	7
Elevator	-	0	0
TCP protocol	-	1	10
Telephone	-	5	53
London Ambulance System	-	5	8491
Simple Arbiter	-	TO	406
Prioritized Protocol	-	TO	8770
Round Robin Arbiter	-	TO	152
Load Balancer	-	TO	6578
Lift Controller	-	TO	2853
AMBA	-	TO	7541

Genetic Algorithm Configuration

Case Study	Spec size	Pop size	Chrom size	Generations
Achieve-Avoid	**3**	**100**	**20**	**50**
Retraction 1	**2**	**100**	**20**	**50**
Retraction 2	**2**	**100**	**20**	**50**
RailRoadCrossingSyste	**4**	**100**	**20**	**50**
MinePump	**3**	**100**	**20**	**50**
ATM	**3**	**100**	**20**	**50**
Elevator	**2**	**100**	**20**	**50**
TCP protocol	**2**	**100**	**20**	**50**
Telephone	**5**	**500**	**50**	**50**
London Ambulance	**5**	**200**	**50**	**50**
Simple Arbiter	**7**	**100**	**50**	**50**
Prioritized Protocol	**7**	**100**	**50**	**50**
Round Robin Arbiter	**9**	**100**	**20**	**50**
Load Balancer	**11**	**200**	**50**	**50**
Lift Controller	**21**	**100**	**50**	**50**
AMBA	**27**	**100**	**50**	**50**