Hybrid Satisfiability Methods for the Inference of Boolean Regulations Controlling Metabolic Networks

Reviewers:

François Fages – Centre INRIA Saclay
Simon de Givry – INRAE – Toulouse

Examiners:

Emmanuelle Becker – Université de Rennes
Misbah Razzaq – INRAE – Tours
Laurent Tournier – INRAE – Jouy-en-Josas

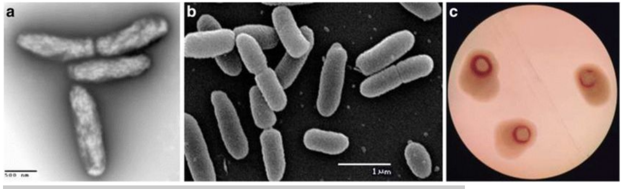
Supervisor:

Anne Siegel – CNRS – Rennes

Co-Supervisor:

Loïc Paulevé – CNRS – Bordeaux

Context



R. solanacearum highly monitored pathogen in agronomy [Puigvert *et al.*, 2016]

Understanding cells is a major challenge in many fields
e.g. *agronomy and health: better understand disease*

Computer science is essential to handle the large quantity of biological data

- **Extract biological knowledge** from data
- Aims at providing **decision-aid tools** for biologists
  e.g. therapeutic target identification, experimental planification

System biology: consider living organisms as interconnected systems
Integrating knowledge and biological data into models

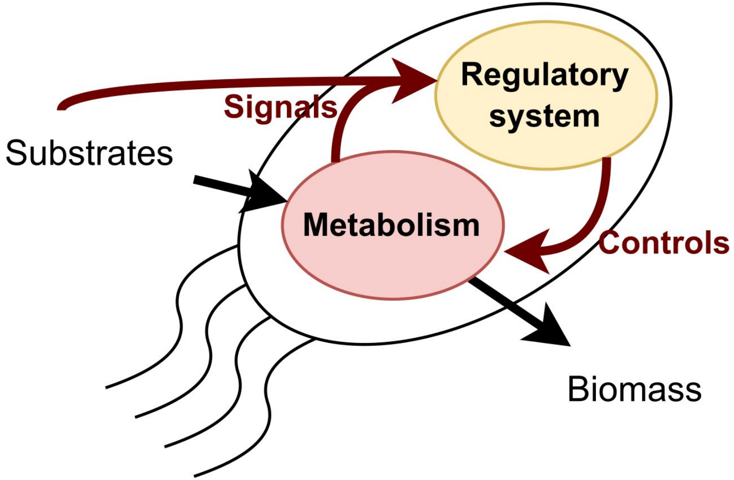
> **Thesis' subject at the interface of system biology, knowledge representation, and formal methods**

State of the Art:

Cells’ multi-scale modelings

Cells: hybrid multi-scale structures

Composed of thousands of ***interconnected*** *chemical processes*
Occurring at different ***scales***

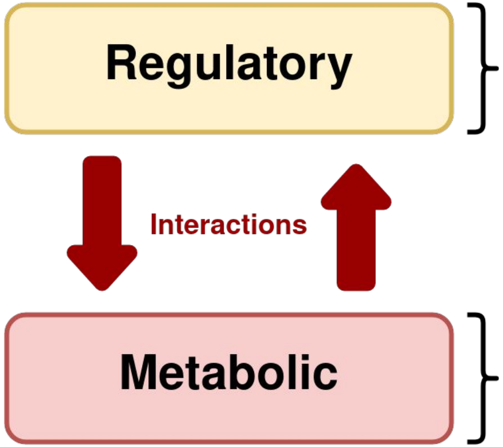


1. **Metabolic scale**
   Chemicals reactions converting substrates to energy and biomass

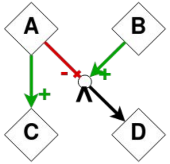
2. **Regulatory scale**
   Rules constraining the metabolism to adapt itself to its environment

Two scales of interest: metabolic and regulatory

Overview of modeling formalisms



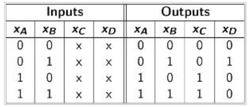
Discrete dynamics [Thomas, 1973]



Model

$$
\begin{aligned}
f_A(x_A, x_B, x_C, x_D) &= x_A \\
f_B(x_A, x_B, x_C, x_D) &= x_B \\
f_C(x_A, x_B, x_C, x_D) &= x_A \\
f_D(x_A, x_B, x_C, x_D) &= \neg x_A \wedge x_B
\end{aligned}
$$

Logical rules
(Boolean network)



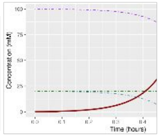
Inputs				Outputs			
x_A	x_B	x_C	x_D	x_A	x_B	x_C	x_D
0	0	x	x	0	0	0	0
0	1	x	x	0	1	0	1
1	0	x	x	1	0	1	0
1	1	x	x	1	1	1	0

Truth table (simulation)

Steady-states approximation [Orth *et al.*, 2010]

$$
\begin{aligned}
&\text{maximise} \quad v_{\text{Growth}} \\
&\text{such that:} \quad S \cdot v = 0 \\
&\qquad\qquad\quad l_r \cdot x_r \leq v_r \leq u_r \cdot x_r \quad \forall r \in \text{reactions}
\end{aligned}
$$

Regulatory flux balance analysis (rFBA)

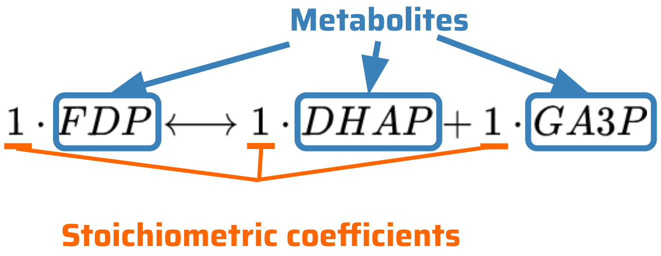


Metabolic traces

Two scales model based on different paradigms and formalisms

Actors

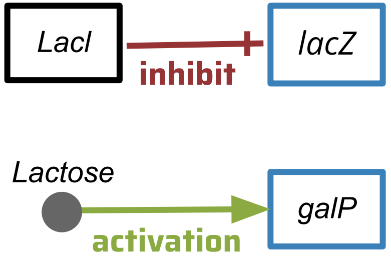
Metabolic scale


Metabolites
$1 \cdot \boxed{FDP} \longleftrightarrow 1 \cdot \boxed{DHAP} + 1 \cdot \boxed{GA3P}$
Stoichiometric coefficients

Metabolites are consumed by reactions to produce other metabolites

Focus on reactions activity rates

Regulatory scale



Protein LacI inhibit the gene lacZ

Lactose activate the gene galP

Components interact to activate or inhibit genes

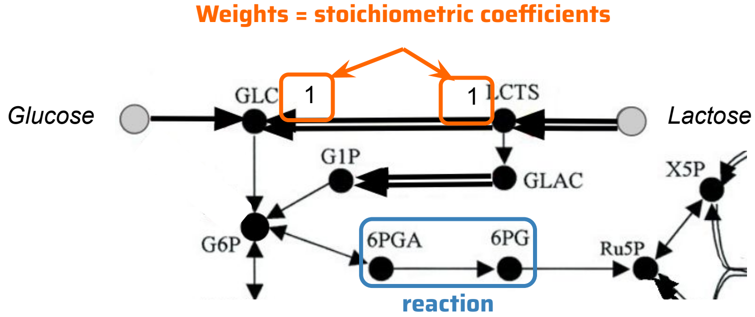
Focus on interactions

Inspired by [Covert and Palsson, 2002]

Structure

Metabolic scale

Metabolic networks

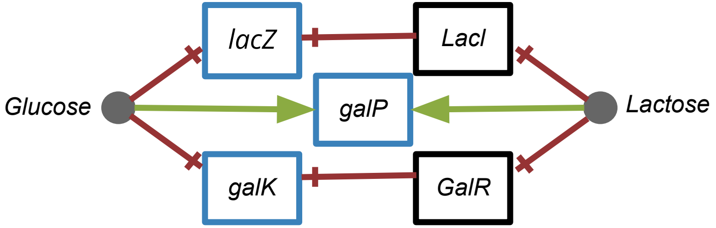

Weights = stoichiometric coefficients

reaction

Weighted hypergraph

Regulatory scale

Interactions graph



Boolean network [Thomas, 1973]

Logical combination of interactions

$$f_{\text{lacZ}}(x) = \neg x_{\text{Glucose}} \wedge \neg x_{\text{LacI}} \quad f_{\text{LacI}}(x) = \neg x_{\text{Lactose}}$$

$$f_{\text{galP}}(x) = x_{\text{Glucose}} \vee x_{\text{Lactose}} \quad f_{\text{galK}}(x) = \neg x_{\text{Glucose}} \wedge \neg x_{\text{GalR}}$$

$$f_{\text{GalR}}(x) = \neg x_{\text{Lactose}}$$

Set of logical rules paired with an directed labeled graph

Inspired by [Covert and Palsson, 2002]

Dynamics

Metabolic scale

Flux balance analysis[1] (FBA) [Orth *et al.*, 2010]

$$\begin{aligned} \text{maximize} \quad & \boxed{v_{\text{Growth}}} \\ \text{such that:} \quad & S \cdot v = 0 \\ & l_r \leq v_r \leq u_r \quad \forall r \in \text{reactions} \end{aligned}$$

Bacteria growth maximization
[Feist and Palsson, 2010]

Based on heuristics: growth optimization + steady-state

Flux-based dynamics

Regulatory scale

Glucose	Lactose	lacZ	galKTEU	LacI	GalR
1	0	1	0	0	1

$$\begin{aligned} f_{\text{lacZ}}(x) &= \neg x_{\text{Glucose}} \wedge \neg x_{\text{LacI}} \\ f_{\text{galP}}(x) &= x_{\text{Glucose}} \vee x_{\text{Lactose}} \\ f_{\text{GalR}}(x) &= \neg x_{\text{Lactose}} \\ f_{\text{LacI}}(x) &= \neg x_{\text{Lactose}} \\ f_{\text{galK}}(x) &= \neg x_{\text{Glucose}} \wedge \neg x_{\text{GalR}} \end{aligned}$$

Glucose	Lactose	lacZ	galKTEU	LacI	GalR
1	0	0	0	1	1

Discrete dynamics [Thomas, 1973]

Various update semantics

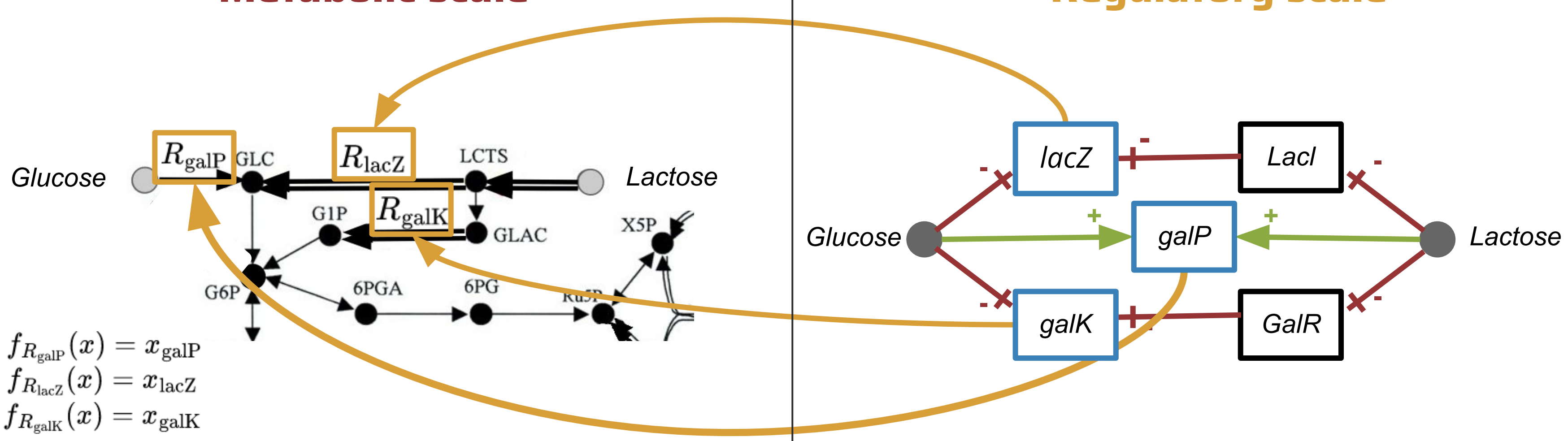
Scale dynamics are based on different paradigms

No straightforward formalism to encompass them

Coupling the scales

Metabolic scale

Regulatory scale



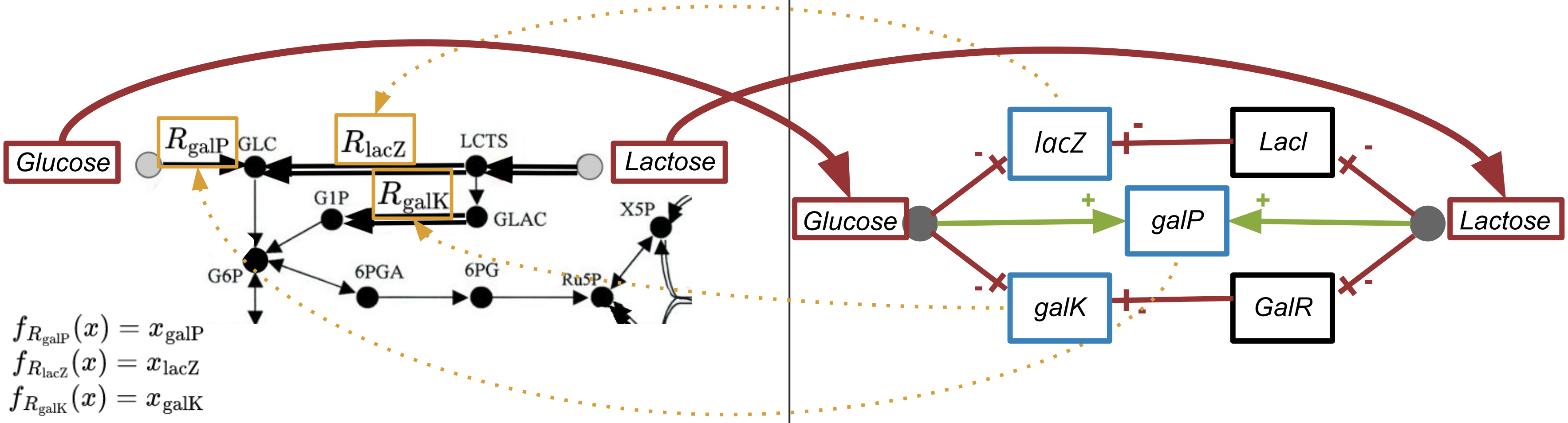
$$f_{R_{\mathrm{galP}}}(x) = x_{\mathrm{galP}}$$
$$f_{R_{\mathrm{lacZ}}}(x) = x_{\mathrm{lacZ}}$$
$$f_{R_{\mathrm{galK}}}(x) = x_{\mathrm{galK}}$$

Regulatory controls:
Regulatory states impact reactions

Metabolic feedback:
Reactions + metabolites impact regulatory states

Interconnected scales through regulatory controls and metabolic feedback
Simulating the coupled dynamics through regulatory Flux Balance Analysis (rFBA) [Covert *et al.*, 2001]

Coupling the scales



$$f_{R_{\mathrm{galP}}}(x) = x_{\mathrm{galP}}$$
$$f_{R_{\mathrm{lacZ}}}(x) = x_{\mathrm{lacZ}}$$
$$f_{R_{\mathrm{galK}}}(x) = x_{\mathrm{galK}}$$

Regulatory controls:
Regulatory states impact reactions

Metabolic feedback:
Reactions + metabolites impact regulatory states

Interconnected scales through regulatory controls and metabolic feedback

Simulating the coupled dynamics through regulatory Flux Balance Analysis (rFBA) **[Covert *et al.*, 2001]**

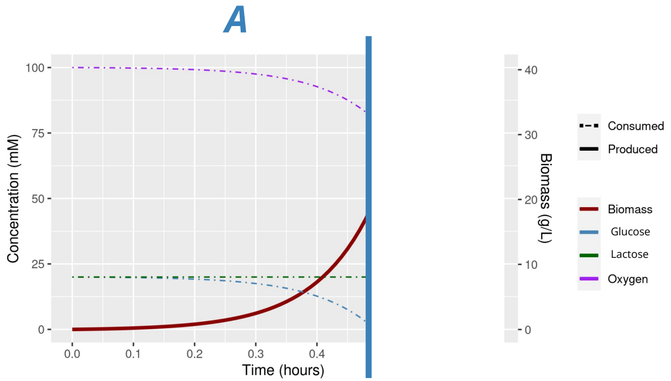
Example of controlled induced behavior: diauxic shift[1]

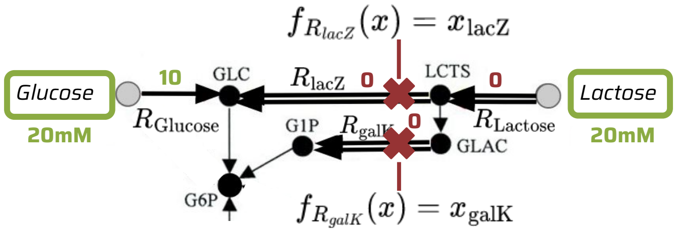
Successives growth phases on different mediums
Controlled by the regulatory scale

needed to import lactose

	Glucose	Lactose	lacZ	galK	LacI	GalR	
A	20mM	20mM	0	0	0	0	Growth on Glucose

rFBA simulation made with FlexFlux [Marmiesse *et al.*, 2015]





Phase A: **lactose** ***could not be imported due to regulatory rules***

[1] J. Monod, **Annales de l'Institut Pasteur**, 1942

Example of controlled induced behavior: diauxic shift[1]

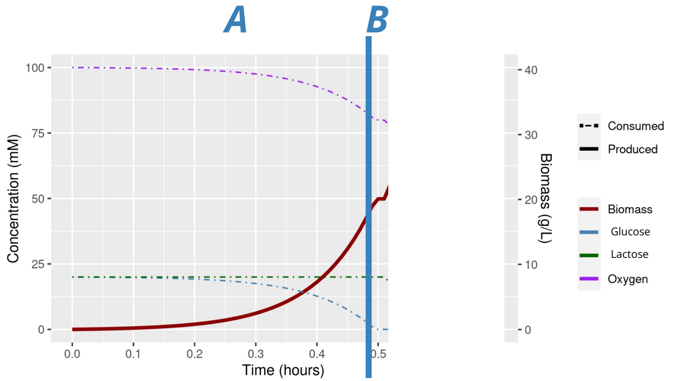
Successives growth phases on different mediums
Controlled by the regulatory scale

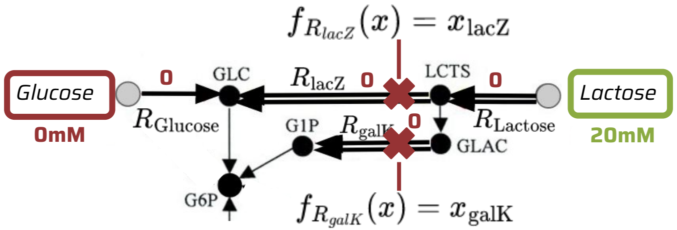
needed to import lactose

	Glucose	Lactose	lacZ	galK	LacI	GalR	
A	20mM	20mM	0	0	0	0	Growth on Glucose
B	0mM	20mM	0	0	0	0	No growth: cannot import Lactose

↓ **Metabolic feedback: glucose is depleted**

rFBA simulation made with FlexFlux **[Marmiesse *et al.*, 2015]**



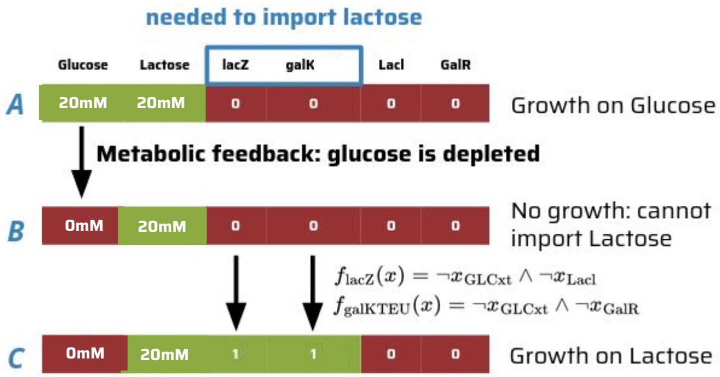


Phase B: ***regulatory mechanisms are slow and need time to react to*** **glucose** ***depletion***

[1] J. Monod, **Annales de l'Institut Pasteur**, 1942

Example of controlled induced behavior: diauxic shift[1]

Successives growth phases on different mediums
Controlled by the regulatory scale



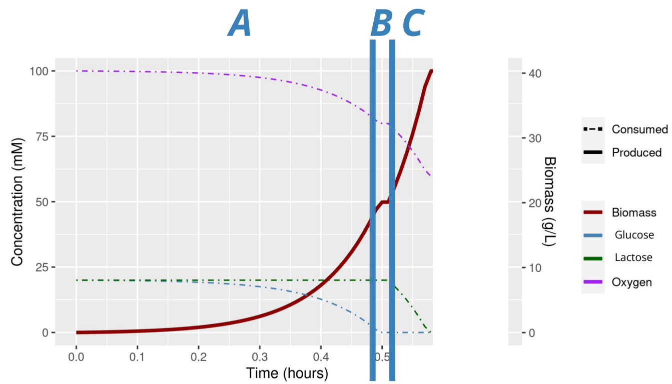
needed to import lactose

	Glucose	Lactose	lacZ	galK	LacI	GalR	
A	20mM	20mM	0	0	0	0	Growth on Glucose
B	0mM	20mM	0	0	0	0	No growth: cannot import Lactose
C	0mM	20mM	1	1	0	0	Growth on Lactose

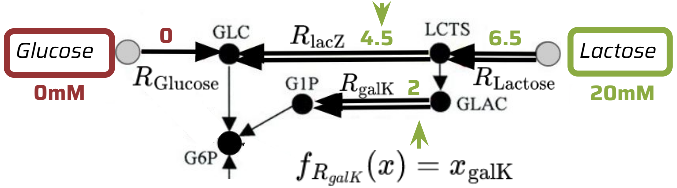
Metabolic feedback: glucose is depleted

$$f_{\text{lacZ}}(x) = \neg x_{\text{GLCxt}} \wedge \neg x_{\text{LacI}}$$
$$f_{\text{galKTEU}}(x) = \neg x_{\text{GLCxt}} \wedge \neg x_{\text{GalR}}$$

rFBA simulation made with FlexFlux **[Marmiesse *et al.*, 2015]**



$$f_{R_{lacZ}}(x) = x_{\text{lacZ}}$$



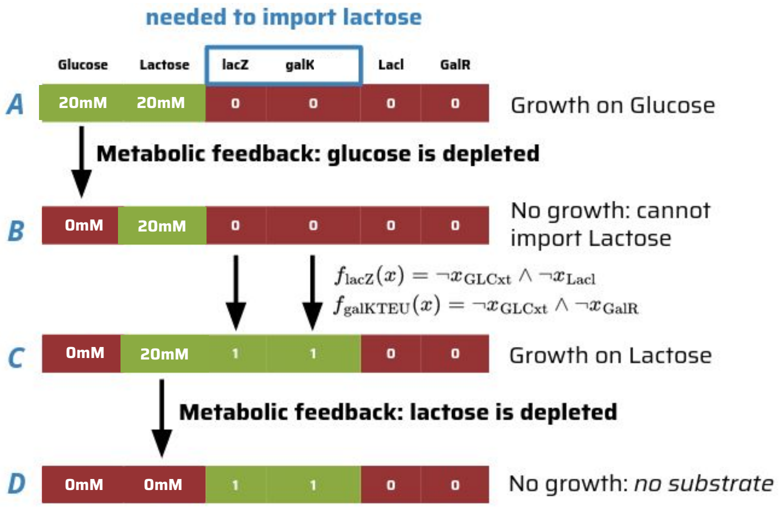
$$f_{R_{galK}}(x) = x_{\text{galK}}$$

Phase C: lacZ *and* galKTEU *states are updated allowing to import lactose*

[1] J. Monod, **Annales de l'Institut Pasteur**, 1942

Example of controlled induced behavior: diauxic shift[1]

Successives growth phases on different mediums
Controlled by the regulatory scale



needed to import lactose

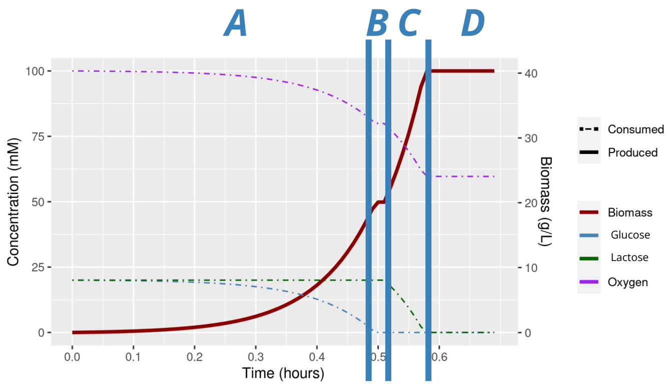
	Glucose	Lactose	lacZ	galK	LacI	GalR	
A	20mM	20mM	0	0	0	0	Growth on Glucose
B	0mM	20mM	0	0	0	0	No growth: cannot import Lactose
C	0mM	20mM	1	1	0	0	Growth on Lactose
D	0mM	0mM	1	1	0	0	No growth: *no substrate*

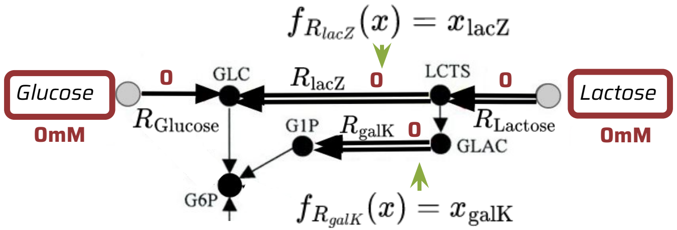
Metabolic feedback: glucose is depleted

$$f_{\mathrm{lacZ}}(x) = \neg x_{\mathrm{GLCxt}} \wedge \neg x_{\mathrm{LacI}}$$
$$f_{\mathrm{galKTEU}}(x) = \neg x_{\mathrm{GLCxt}} \wedge \neg x_{\mathrm{GalR}}$$

Metabolic feedback: lactose is depleted

rFBA simulation made with FlexFlux **[Marmiesse *et al.*, 2015]**



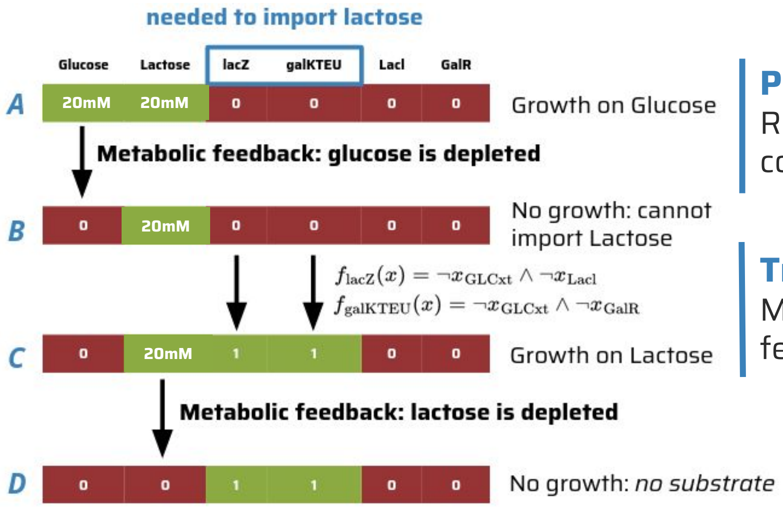


$$f_{R_{lacZ}}(x) = x_{\mathrm{lacZ}}$$
$$f_{R_{galK}}(x) = x_{\mathrm{galK}}$$

Phase D: ***no carbon sources to allow growth***

[1] J. Monod, **Annales de l'Institut Pasteur**, 1942

What we learned



needed to import lactose

	Glucose	Lactose	lacZ	galKTEU	LacI	GalR	
A	20mM	20mM	0	0	0	0	Growth on Glucose
B	0	20mM	0	0	0	0	No growth: cannot import Lactose
C	0	20mM	1	1	0	0	Growth on Lactose
D	0	0	1	1	0	0	No growth: *no substrate*

Metabolic feedback: glucose is depleted

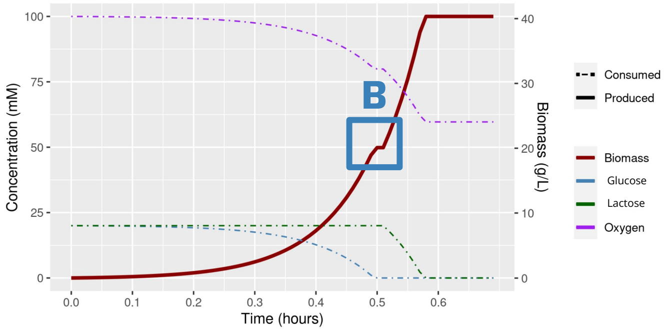
$f_{\text{lacZ}}(x) = \neg x_{\text{GLCxt}} \wedge \neg x_{\text{LacI}}$

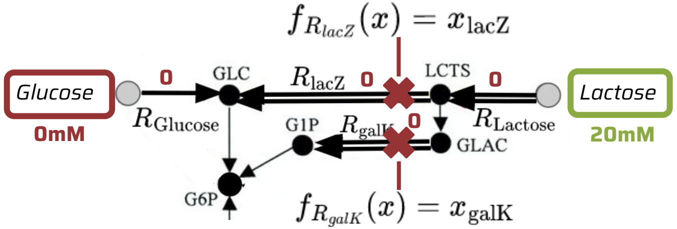
$f_{\text{galKTEU}}(x) = \neg x_{\text{GLCxt}} \wedge \neg x_{\text{GalR}}$

Metabolic feedback: lactose is depleted

Phases A / B: Regulatory controls

Transition B → C: Metabolic feedback





$f_{R_{lacZ}}(x) = x_{\text{lacZ}}$

$f_{R_{galK}}(x) = x_{\text{galK}}$

Regulation has impacts on growth

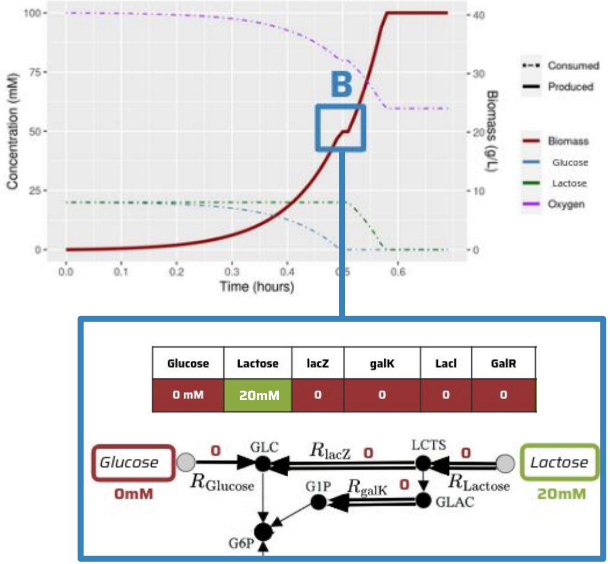
Indirect observation of the regulation on phase B

Regulation impacts on the metabolism are hard to detect

Our challenge

$$
\begin{aligned}
f_{\mathrm{lacZ}}(x) &= \\
f_{\mathrm{galP}}(x) &= \\
f_{\mathrm{GalR}}(x) &= \\
f_{\mathrm{LacI}}(x) &= \\
f_{\mathrm{galK}}(x) &= \\
f_{R_{lacZ}}(x) &= \\
f_{R_{galK}}(x) &=
\end{aligned}
$$

???



Glucose	Lactose	lacZ	galK	LacI	GalR
0 mM	20mM	0	0	0	0

Thesis' objective: can we infer regulatory control of metabolism?

Define methods to infer Boolean regulatory rules controlling the metabolism from metabolic time series observations

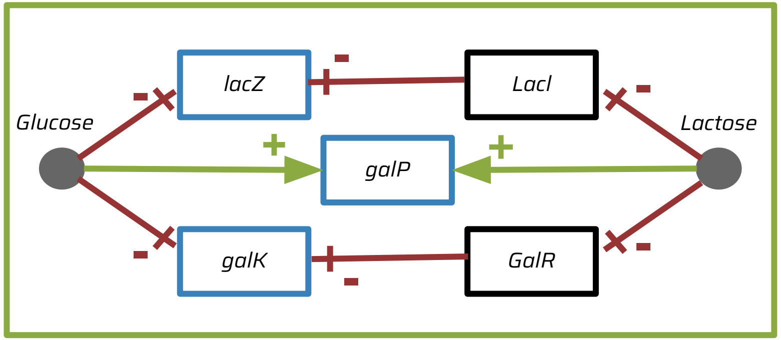
State of the Art:

Inference of Boolean regulatory rules

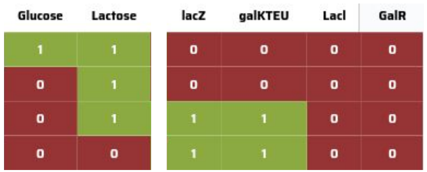
Inference of Boolean networks in the literature: overview

Input:

Interaction graph: define a search space



Observations



Glucose	Lactose	lacZ	galKTEU	LacI	GalR
1	1	0	0	0	0
0	1	0	0	0	0
0	1	1	1	0	0
0	0	1	1	0	0

Gene expression on different experimental conditions

State-of-the-art methods consider direct impacts of the regulation

Output:

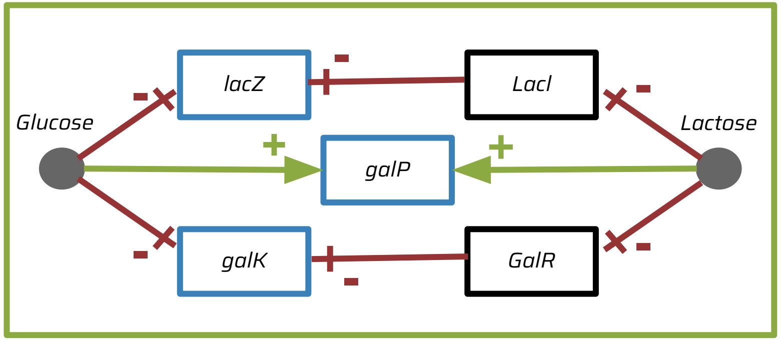
Optimal Boolean networks **in the search space compatible with the observations**

e.g. optimality criteria: network size, observation matching

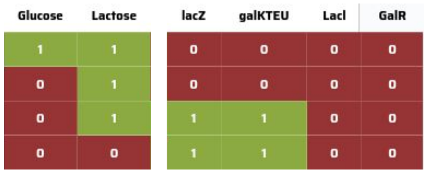
Inference of Boolean networks in the literature: overview

Input:

Interaction graph: define a search space



Observations



Glucose	Lactose	lacZ	galKTEU	LacI	GalR
1	1	0	0	0	0
0	1	0	0	0	0
0	1	1	1	0	0
0	0	1	1	0	0

Gene expression on different experimental conditions

State-of-the-art methods consider direct impacts of the regulation

Output:

Optimal Boolean networks **in the search space compatible with the observations**

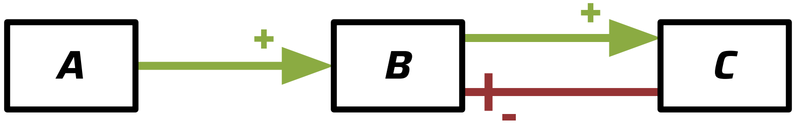
e.g. optimality criteria: network size, observation matching

Methods differs from their “compatibility” + “optimality” criteria

Search space

Set of Boolean networks compatible with an interaction graph

Example



$$f_A(x) = 0$$
$$f_A(x) = 1$$
$$f_A(x) = x_A$$
$$f_A(x) = \neg x_B$$
$$f_A(x) = x_A \vee \neg x_B$$
$$f_A(x) = x_A \wedge \neg x_B$$

6 rules

Regulatory rule of ***B*** can only depends on:

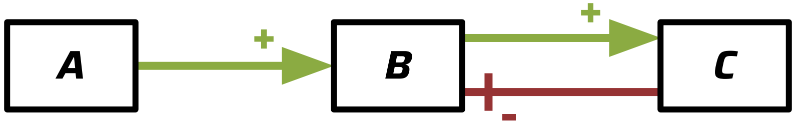
- ➔ constant value 0 or 1
- ➔ activation of *A*
- ➔ inhibition of *C*

Rules are logical combinations of the interactions

Search space

Set of Boolean networks compatible with an interaction graph

Example



$f_A(x) = 0$

$f_A(x) = 1$

2 rules

$f_A(x) = 0$

$f_A(x) = 1$

$f_A(x) = x_A$

$f_A(x) = \neg x_B$

$f_A(x) = x_A \vee \neg x_B$

$f_A(x) = x_A \wedge \neg x_B$

6 rules

$f_C(x) = 0$

$f_C(x) = 1$

$f_C(x) = x_B$

3 rules

Regulatory rule of ***B*** can only depends on:

- ➔ constant value 0 or 1
- ➔ activation of *A*
- ➔ inhibition of *C*

36 Boolean networks compatible
36 = 2 x 6 x 3

Number of compatible Boolean networks is doubly exponential in the number of interactions

Methods that infer regulatory rules

Methods		Paradigm	Observations	Semantics	Inferred models
CellNOptR	[Terfve *et al.*, 2012]	**Constraint Programming** Mixed Integer Linear Programming	**Steady-state**	Fixpoint - synchronous	**Boolean networks** **Optimizing:** **- size** **- data fitting** **Ignore:** **- metabolic feedback** **- regulatory controls** **Regulatory**
CASPO	[Videla *et al.*, 2017]	**Constraint Programming** Combinatorial optimization problem			
CaspoTS	[Ostrowski *et al.*, 2016]		**Time series**	Meta-Boolean network	
BoNesis	[Chevalier *et al.*, 2020]			Most Permissive	
ASKEed	[Vaginay *et al.*, 2021]		**Time series** *Multivariate*	Reachability - (a)synchronous	
CGA-BNI	[Trinh and Kwon, 2021]	**Genetic algorithm**	**Steady-state**	Fixpoint - synchronous	
SgpNet	[Gao *et al.*, 2020]		**Time series**	Reachability - asynchronous	
Gapore	[Liu *et al.*, 2021]			Reachability - synchronous	
NNBNI	[Barman and Kwon, 2020]	**Neural network** Supervised			

Methods that infer regulatory rules

Methods		Paradigm	Observations	Semantics	Inferred models
CellNOptR	[Terfve *et al.*, 2012]	**Constraint Programming** Mixed Integer Linear Programming	**Steady-state**	Fixpoint - synchronous	**Boolean networks** **Optimizing:** **- size** **- data fitting** **Ignore:** **- metabolic feedback** **- regulatory controls** **Regulatory**
CASPO	[Videla *et al.*, 2017]	**Constraint Programming** Combinatorial optimization problem			
CaspoTS	[Ostrowski *et al.*, 2016]		**Time series**	Meta-Boolean network	
BoNesis	[Chevalier *et al.*, 2020]			Most Permissive	
ASKEed	[Vaginay *et al.*, 2021]		**Time series** *Multivariate*	Reachability - (a)synchronous	
CGA-BNI	[Trinh and Kwon, 2021]	**Genetic algorithm**	**Steady-state**	Fixpoint - synchronous	
SgpNet	[Gao *et al.*, 2020]		**Time series**	Reachability - asynchronous	
Gapore	[Liu *et al.*, 2021]			Reachability - synchronous	
NNBNI	[Barman and Kwon, 2020]	**Neural network** Supervised			

Constraint programming-based approaches allow for curated interaction graphs

Methods that infer regulatory rules

Methods		Paradigm	Observations	Semantics	Inferred models
CellNOptR	[Terfve *et al.*, 2012]	**Constraint Programming** Mixed Integer Linear Programming	**Steady-state**	Fixpoint - synchronous	**Boolean networks** **Optimizing:** **- size** **- data fitting** **Ignore:** **- metabolic feedback** **- regulatory controls** **Regulatory**
CASPO	[Videla *et al.*, 2017]	**Constraint Programming** Combinatorial optimization problem			
CaspoTS	[Ostrowski *et al.*, 2016]		**Time series**	Meta-Boolean network	
BoNesis	[Chevalier *et al.*, 2020]			Most Permissive	
ASKEed	[Vaginay *et al.*, 2021]		**Time series** *Multivariate*	Reachability - (a)synchronous	
CGA-BNI	[Trinh and Kwon, 2021]	**Genetic algorithm**	**Steady-state**	Fixpoint - synchronous	
SgpNet	[Gao *et al.*, 2020]		**Time series**	Reachability - asynchronous	
Gapore	[Liu *et al.*, 2021]			Reachability - synchronous	
NNBNI	[Barman and Kwon, 2020]	**Neural network** Supervised			

Limits: do not use metabolic observations + ignore feedback and controls effects

Methods that infer regulatory rules

Methods		Paradigm	Observations	Semantics	Inferred models
CellNOptR	[Terfve *et al.*, 2012]	**Constraint Programming** Mixed Integer Linear Programming	**Steady-state**	Fixpoint - synchronous	**Boolean networks** **Optimizing:** **- size** **- data fitting** **Ignore:** **- metabolic feedback** **- regulatory controls** **Regulatory**
CASPO	[Videla *et al.*, 2017]	**Constraint Programming** Combinatorial optimization problem			
CaspoTS	[Ostrowski *et al.*, 2016]		**Time series**	Meta-Boolean network	
BoNesis	[Chevalier *et al.*, 2020]			Most Permissive	
ASKEed	[Vaginay *et al.*, 2021]		**Time series** *Multivariate*	Reachability - (a)synchronous	
CGA-BNI	[Trinh and Kwon, 2021]	**Genetic algorithm**	**Steady-state**	Fixpoint - synchronous	
SgpNet	[Gao *et al.*, 2020]		**Time series**	Reachability - asynchronous	
Gapore	[Liu *et al.*, 2021]			Reachability - synchronous	
NNBNI	[Barman and Kwon, 2020]	**Neural network** Supervised			

Capturing metabolic feedback requires combining linear and combinatorial optimization

Thesis' contributions

> **Thesis' objective:**
> Formalize and solve the problem of the inference of regulatory rules ***that controls metabolic networks*** from observations and ***curated interaction graph***

Contributions' outline:

1. **Formalization:** of our inference problem as a *combinatorial optimization problem modulo quantified linear constraints (OPT+qLP)*
2. **Solving method:** a generic workflow to address *OPT+qLP*
3. **Benchmark and validation:** application to a benchmark based on *Escherichia coli*

Contribution 1:

Formalization

Formalization of our inference problem

$$
\begin{aligned}
&\text{minimize} \quad f_{\text{obj}}(x) \\
&\text{such that} \\
&\quad \bigwedge_{\alpha} \boxed{c_\alpha(x)} \\
&\quad \wedge \quad \bigwedge_{\beta} c_\beta(x) \vee \boxed{g_\beta(y) \leq 0} \\
&\quad \wedge \quad \forall z \in \mathbb{R}^p, \left(\bigwedge_{\gamma} c_\gamma(x) \vee g_\gamma(z) \leq 0 \right) \implies \left(\bigwedge_{\zeta} c_\zeta(x) \vee g_\zeta(z) \leq 0 \right) \\
&\text{with } x \in \mathbb{B}^n,\ y \in \mathbb{R}^m
\end{aligned}
$$

$c(x)$: $\bigvee_i x_i \vee \bigvee_j \neg x_j$

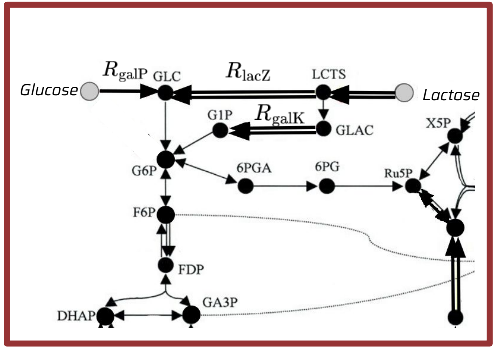
$g(y)$: linear function

Combinatorial optimization problem modulo quantified linear constraints (OPT+qLP)

Inference of Boolean networks controlling the metabolism

Input:

Metabolic network



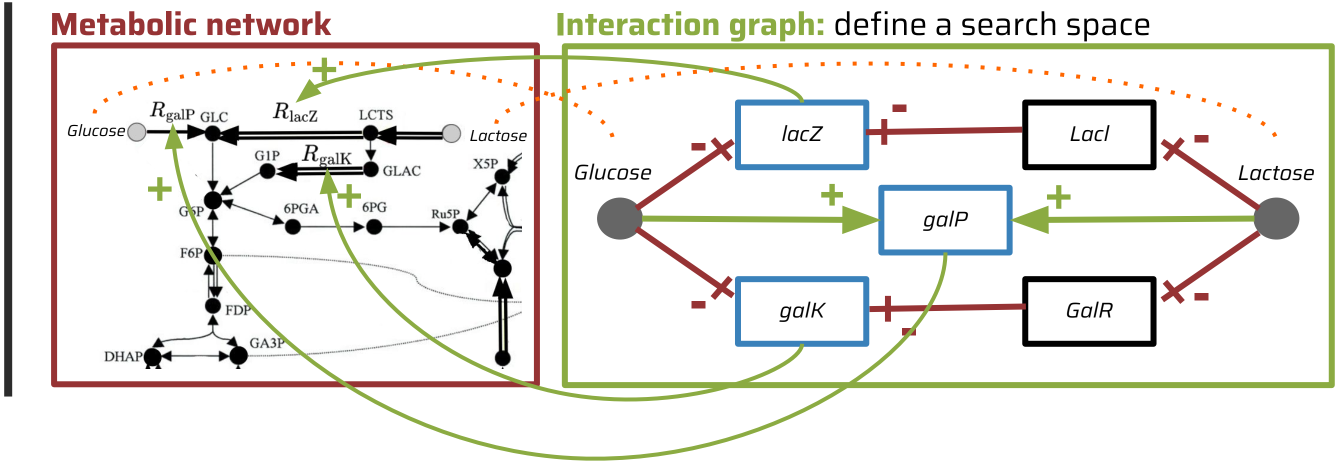
Metabolic network is an input

Standard protocol to reconstruct [Thiele *et al.*, 2010]

Public databases with high quality curated metabolic networks e.g. *BiGG* - [King *et al.*, 2015]

Inference of Boolean networks controlling the metabolism

Input:



A curated interaction graph

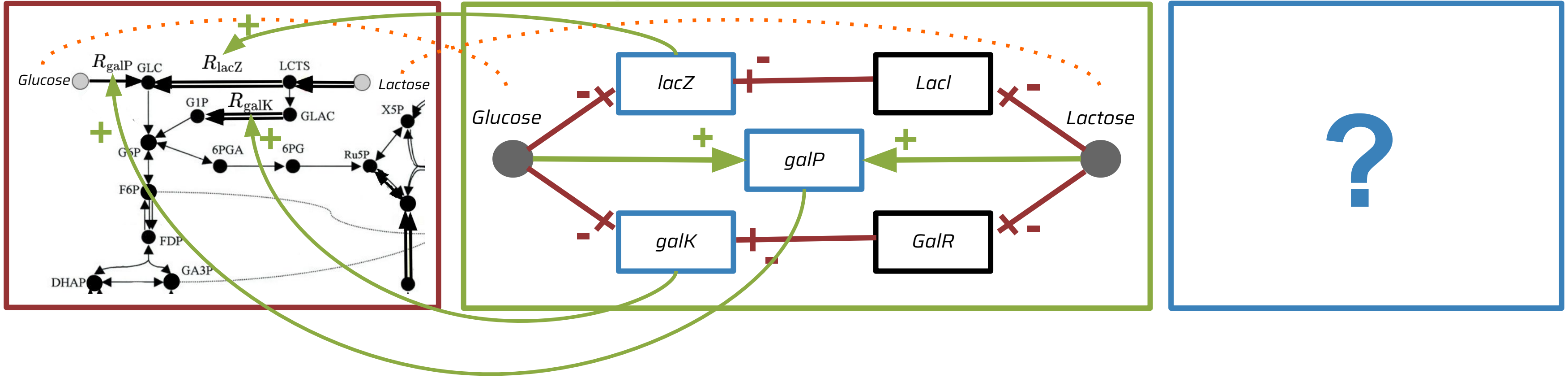
Set of manually selected interactions

Accounting for all the interactions between the regulatory and metabolic scales

Inference of Boolean networks controlling the metabolism

Input:

Metabolic network | **Interaction graph:** define a search space | **Observations**



Time series observations

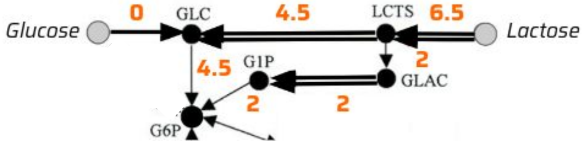
Direct observations:

- ➔ **Transcriptomics** qualitative
  Gene expression data
  Reaction and metabolite state

Glucose	Lactose	lacZ	galK	LacI	GalR	R_lacZ	R_galK
0	1	1	1	0	0	1	1

Indirect observations:

- ➔ **Fluxomics** quantitative
  Rates of reactions activity



- ➔ **Kinetics** quantitative
  Substrate concentrations

Glucose	Lactose	...
0 mM	20 mM	...

+ Growth rate = 1.12

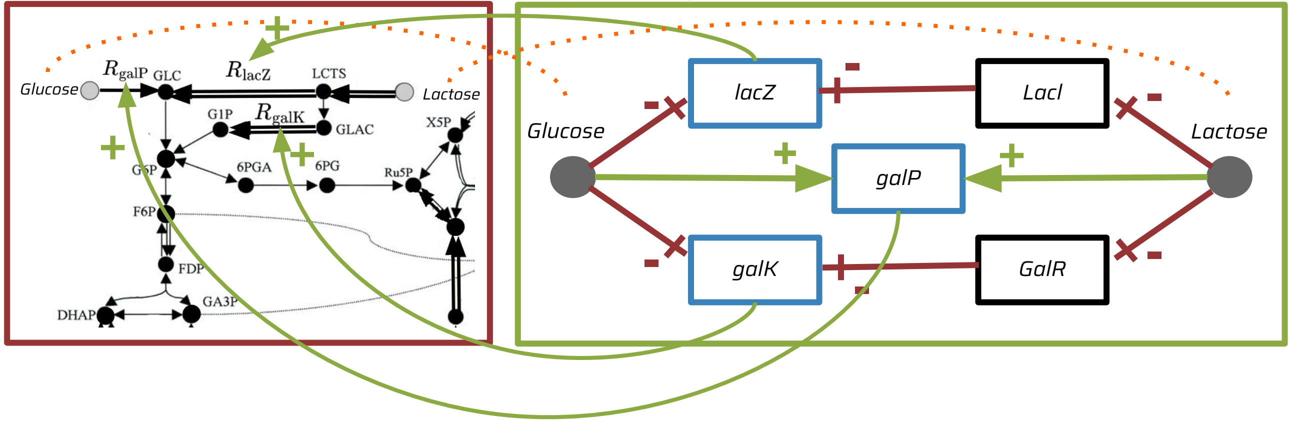
3 data types of interest to infer regulatory rules controlling metabolic networks

Inference of Boolean networks controlling the metabolism

Input:

Metabolic network

Interaction graph: define a search space



***Time series* observations**

Direct observations:

- transcriptomics

Indirect observations:

- kinetics
- fluxomics

Output:

> ***Optimal*** Boolean networks in the search space with a ***trace compatible with the observations***

General form of the inference problem

$c(x) : \bigvee_i x_i \vee \bigvee_j \neg x_j$

$g(y)$: linear function

$$\text{minimize} \quad f_{\text{obj}}(x)$$

Optimization criteria

$$\text{such that}$$

$$\bigwedge_\alpha c_\alpha(x)$$

Search space

$$\wedge \quad \bigwedge_\beta c_\beta(x) \vee g_\beta(y) \leq 0$$

$$\wedge \quad \forall z \in \mathbb{R}^p, \left(\bigwedge_\gamma c_\gamma(x) \vee g_\gamma(z) \leq 0 \right) \implies \left(\bigwedge_\zeta c_\zeta(x) \vee g_\zeta(z) \leq 0 \right)$$

$$\text{with } x \in \mathbb{B}^n,\ y \in \mathbb{R}^m$$

Output:

> ***Optimal*** Boolean networks in the search space with a ***trace compatible with the observations***

Regulated metabolic state

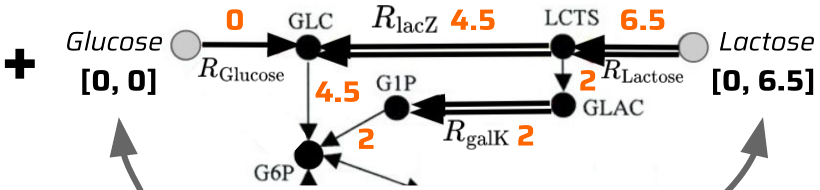
1. **Regulatory state**
   Boolean regulatory state of each element

Glucose	Lactose	lacZ	galK	LacI	GalR	R_lacZ	R_galK
0	1	1	1	0	0	1	1

2. **Metabolic state**
   The metabolic activity of each reaction, such that:

$$\begin{aligned} \text{maximize} \quad & v_{\text{Growth}} \\ \text{such that:} \quad & S \cdot v = 0 \\ & l_r \le v_r \le u_r \quad \forall r \in \text{reactions} \\ & v_r = 0 \quad \forall r \in \text{inhibited reactions} \end{aligned}$$

Growth rate = 1.12



3. **Substrate state**
   External metabolite concentrations

Glucose	Lactose	...
0 mM	20 mM	...

Use to compute thermodynamic bounds

States are composed of 3 layers as for the observations

rFBA states transition

Regulatory flux balance analysis (rFBA) [Covert *et al.*, 2001]

rFBA transition:

1. Update the **regulatory state**
   Synchronous update of the regulatory rules

2. Update the **metabolic state**
   Solve the FBA equations:

$$\begin{array}{ll} \text{maximize} & v_{\text{Growth}} \\ \text{such that:} & S \cdot v = 0 \\ & l_r \leq v_r \leq u_r \quad \forall r \in \text{reactions} \\ & v_r = 0 \quad \forall r \in \text{inhibited reactions} \end{array}$$

3. Update the **substrate state**

> **Successive updates of the regulatory (*discrete*) and metabolic (*linear*) states**

rFBA states transition

Regulatory flux balance analysis (rFBA) [Covert *et al.*, 2001]

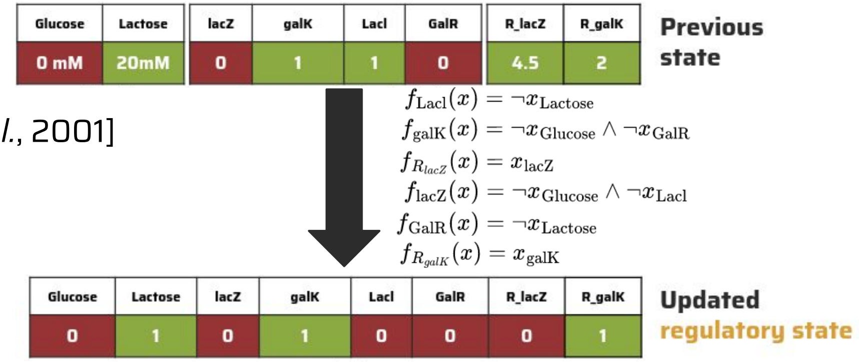
rFBA transition:

1. Update the **regulatory state**
   Synchronous update of the regulatory rules

2. Update the **metabolic state**
   Solve the FBA equations:

$$\begin{array}{ll} \text{maximize} & v_{\text{Growth}} \\ \text{such that:} & S \cdot v = 0 \\ & l_r \leq v_r \leq u_r \quad \forall r \in \text{reactions} \\ & v_r = 0 \quad \forall r \in \text{inhibited reactions} \end{array}$$

3. Update the **substrate state**



Glucose	Lactose	lacZ	galK	LacI	GalR	R_lacZ	R_galK
0 mM	20mM	0	1	1	0	4.5	2

Previous state

$$f_{\text{LacI}}(x) = \neg x_{\text{Lactose}}$$
$$f_{\text{galK}}(x) = \neg x_{\text{Glucose}} \wedge \neg x_{\text{GalR}}$$
$$f_{R_{lacZ}}(x) = x_{\text{lacZ}}$$
$$f_{\text{lacZ}}(x) = \neg x_{\text{Glucose}} \wedge \neg x_{\text{LacI}}$$
$$f_{\text{GalR}}(x) = \neg x_{\text{Lactose}}$$
$$f_{R_{galK}}(x) = x_{\text{galK}}$$

Glucose	Lactose	lacZ	galK	LacI	GalR	R_lacZ	R_galK
0	1	0	1	0	0	0	1

Updated regulatory state

Successive updates of the regulatory (*discrete*) and metabolic (*linear*) states

rFBA states transition

Regulatory flux balance analysis (rFBA) [Covert *et al.*, 2001]

rFBA transition:

1. Update the **regulatory state**
   Synchronous update of the regulatory rules

2. Update the **metabolic state**
   Solve the FBA equations:

$$\begin{aligned} \text{maximize} \quad & v_{\text{Growth}} \\ \text{such that:} \quad & S \cdot v = 0 \\ & l_r \leq v_r \leq u_r \quad \forall r \in \text{reactions} \\ & v_r = 0 \quad \forall r \in \text{inhibited reactions} \end{aligned}$$

3. Update the **substrate state**

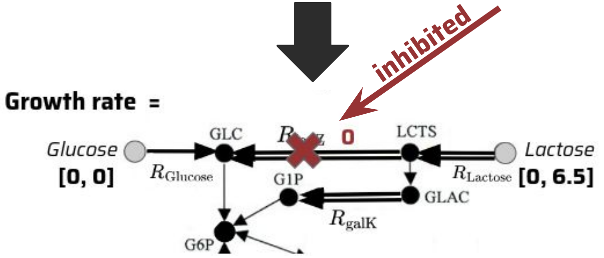
Previous state

Glucose	Lactose	lacZ	galK	LacI	GalR	R_lacZ	R_galK
0 mM	20mM	0	1	1	0	4.5	2

$$f_{\text{LacI}}(x) = \neg x_{\text{Lactose}}$$
$$f_{\text{galK}}(x) = \neg x_{\text{Glucose}} \wedge \neg x_{\text{GalR}}$$
$$f_{R_{lacZ}}(x) = x_{\text{lacZ}}$$
$$f_{\text{lacZ}}(x) = \neg x_{\text{Glucose}} \wedge \neg x_{\text{LacI}}$$
$$f_{\text{GalR}}(x) = \neg x_{\text{Lactose}}$$
$$f_{R_{galK}}(x) = x_{\text{galK}}$$

Updated regulatory state

Glucose	Lactose	lacZ	galK	LacI	GalR	R_lacZ	R_galK
0	1	0	1	0	0	0	1



Updated metabolic state

Successive updates of the regulatory (*discrete*) and metabolic (*linear*) states

rFBA states transition

Regulatory flux balance analysis (rFBA) [Covert *et al.*, 2001]

rFBA transition:

1. Update the **regulatory state**
   Synchronous update of the regulatory rules

2. Update the **metabolic state**
   Solve the FBA equations:

$$\begin{array}{ll} \text{maximize} & v_{\text{Growth}} \\ \text{such that:} & S \cdot v = 0 \\ & l_r \leq v_r \leq u_r \quad \forall r \in \text{reactions} \\ & v_r = 0 \quad \forall r \in \text{inhibited reactions} \end{array}$$

3. Update the **substrate state**

Glucose	Lactose	lacZ	galK	LacI	GalR	R_lacZ	R_galK
0 mM	20mM	0	1	1	0	4.5	2

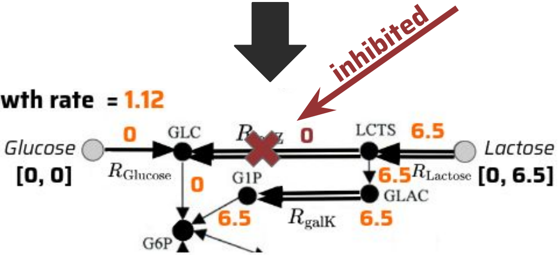
Previous state

$$\begin{aligned} f_{\text{LacI}}(x) &= \neg x_{\text{Lactose}} \\ f_{\text{galK}}(x) &= \neg x_{\text{Glucose}} \wedge \neg x_{\text{GalR}} \\ f_{R_{lacZ}}(x) &= x_{\text{lacZ}} \\ f_{\text{lacZ}}(x) &= \neg x_{\text{Glucose}} \wedge \neg x_{\text{LacI}} \\ f_{\text{GalR}}(x) &= \neg x_{\text{Lactose}} \\ f_{R_{galK}}(x) &= x_{\text{galK}} \end{aligned}$$

Glucose	Lactose	lacZ	galK	LacI	GalR	R_lacZ	R_galK
0	1	0	1	0	0	0	1

Updated regulatory state

Growth rate = 1.12



Updated metabolic state

Successive updates of the regulatory (*discrete*) and metabolic (*linear*) states

rFBA states transition

Regulatory flux balance analysis (rFBA) [Covert *et al.*, 2001]

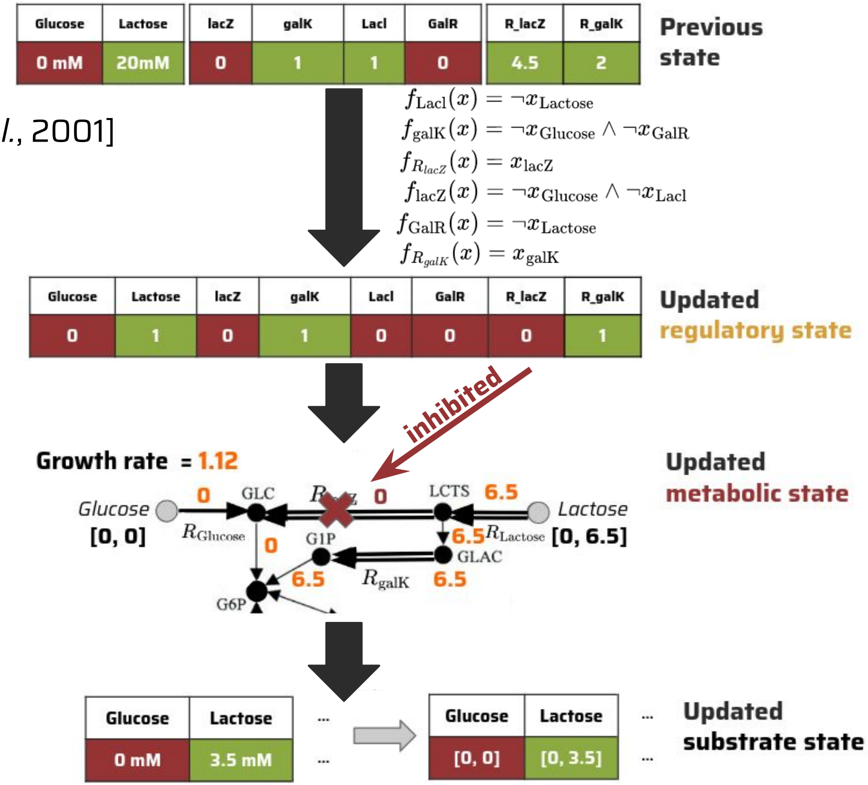
rFBA transition:

1. Update the **regulatory state**
   Synchronous update of the regulatory rules

2. Update the **metabolic state**
   Solve the FBA equations:

$$\begin{aligned} \text{maximize} \quad & v_{\text{Growth}} \\ \text{such that:} \quad & S \cdot v = 0 \\ & l_r \leq v_r \leq u_r \quad \forall r \in \text{reactions} \\ & v_r = 0 \quad \forall r \in \text{inhibited reactions} \end{aligned}$$

3. Update the **substrate state**



Glucose	Lactose	lacZ	galK	LacI	GalR	R_lacZ	R_galK
0 mM	20mM	0	1	1	0	4.5	2

Previous state

$$f_{\mathrm{LacI}}(x) = \neg x_{\mathrm{Lactose}}$$
$$f_{\mathrm{galK}}(x) = \neg x_{\mathrm{Glucose}} \wedge \neg x_{\mathrm{GalR}}$$
$$f_{R_{lacZ}}(x) = x_{\mathrm{lacZ}}$$
$$f_{\mathrm{lacZ}}(x) = \neg x_{\mathrm{Glucose}} \wedge \neg x_{\mathrm{LacI}}$$
$$f_{\mathrm{GalR}}(x) = \neg x_{\mathrm{Lactose}}$$
$$f_{R_{galK}}(x) = x_{\mathrm{galK}}$$

Glucose	Lactose	lacZ	galK	LacI	GalR	R_lacZ	R_galK
0	1	0	1	0	0	0	1

Updated regulatory state

Growth rate = 1.12

Updated metabolic state

Glucose	Lactose	...
0 mM	3.5 mM	...

Glucose	Lactose	...
[0, 0]	[0, 3.5]	...

Updated substrate state

Successive updates of the regulatory (*discrete*) and metabolic (*linear*) states

General form of the inference problem

$c(x): \bigvee_i x_i \vee \bigvee_j \neg x_j$

$g(y)$: linear function

$$\begin{aligned}
&\text{minimize} \quad f_{\text{obj}}(x) \\
&\text{such that} \\
&\quad \bigwedge_\alpha c_\alpha(x) \\
&\quad \wedge \quad \bigwedge_\beta c_\beta(x) \vee g_\beta(y) \leq 0 \\
&\quad \wedge \quad \forall z \in \mathbb{R}^p, \left(\bigwedge_\gamma c_\gamma(x) \vee g_\gamma(z) \leq 0 \right) \implies \left(\bigwedge_\zeta c_\zeta(x) \vee g_\zeta(z) \leq 0 \right) \\
&\text{with } x \in \mathbb{B}^n,\ y \in \mathbb{R}^m
\end{aligned}$$

Optimization criteria

Search space + Regulatory state

Metabolic state

Output:

> ***Optimal*** Boolean networks in the search space with a ***trace compatible with the observations***

State and observation compatibility

1. Regulatory state and **substrate state** are equal to transcriptomics and **kinetics**

Transcriptomics:

Glucose	Lactose	lacZ	galK	LacI	GalR	R_lacZ	R_galK
0	1	-	1	0	-	1	1

=

Regulatory state:

Glucose	Lactose	lacZ	galK	LacI	GalR	R_lacZ	R_galK
0	1	1	1	0	0	1	1

Kinetics:

Glucose	Lactose	...
0 mM	20 mM	...

=

Substrate state:

Glucose	Lactose	...
0 mM	20 mM	...

Observations

Regulated metabolic state

Regulatory state matches with input gene expression data

General form of the compatibility constraints

$c(x): \bigvee_i x_i \vee \bigvee_j \neg x_j$

$g(y)$: linear function

$$\begin{aligned}
&\text{minimize} \quad f_{\text{obj}}(x) \\
&\text{such that} \\
&\quad \bigwedge_\alpha c_\alpha(x)
\end{aligned}$$

Criteria 1: regulatory state and **substrate state** are equal to **transcriptomics** and **kinetics**

$$\begin{aligned}
&\wedge \quad \bigwedge_\beta c_\beta(x) \vee g_\beta(y) \leq 0 \\
&\wedge \quad \forall z \in \mathbb{R}^p, \left(\bigwedge_\gamma c_\gamma(x) \vee g_\gamma(z) \leq 0 \right) \implies \left(\bigwedge_\zeta c_\zeta(x) \vee g_\zeta(z) \leq 0 \right) \\
&\text{with } x \in \mathbb{B}^n,\ y \in \mathbb{R}^m
\end{aligned}$$

Compatibility criteria: logic (1) constraints

State and observation compatibility

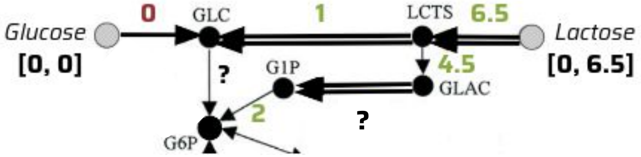
1. **Regulatory state** and **substrate state** are equal to **transcriptomics** and **kinetics**

2. Reaction activity states in the **metabolic state** and **fluxomics** are identical
   + **metabolic state** has the same growth rate as **kinetics**

Kinetics:

Growth rate = 1.12

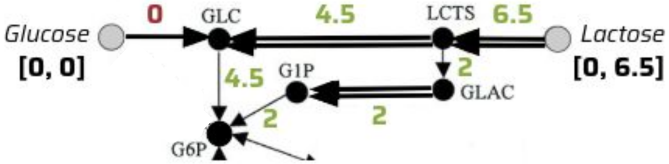
Fluxomics:



Metabolic state:

Growth rate = 1.12

+



Metabolic state's flux distribution matches with the metabolic observations

General form of the compatibility constraints

$c(x): \bigvee_i x_i \vee \bigvee_j \neg x_j$

$g(y)$: linear function

$$
\begin{aligned}
&\text{minimize} \quad f_{\text{obj}}(x) \\
&\text{such that} \\
&\quad \bigwedge_{\alpha} c_\alpha(x) \\
&\quad \wedge \quad \bigwedge_{\beta} c_\beta(x) \vee g_\beta(y) \leq 0 \\
&\quad \wedge \quad \forall z \in \mathbb{R}^p, \left(\bigwedge_{\gamma} c_\gamma(x) \vee g_\gamma(z) \leq 0 \right) \implies \left(\bigwedge_{\zeta} c_\zeta(x) \vee g_\zeta(z) \leq 0 \right) \\
&\text{with } x \in \mathbb{B}^n,\ y \in \mathbb{R}^m
\end{aligned}
$$

Criteria 1: regulatory state and **substrate state** are equal to **transcriptomics** and **kinetics**

Criteria 2: reaction activity states in the **metabolic state** and **fluxomics** are identical + **metabolic state** has the same growth rate as **kinetics**

Compatibility criteria: logic (1) + linear (2) constraints

State and observation compatibility

1. **Regulatory state** and **substrate state** are equal to **transcriptomics** and **kinetics**

2. Reaction activity states in the **metabolic state** and **fluxomics** are identical
   + metabolic state has the same growth rate as **kinetics**

3. All compatible **metabolic states** have a maximum growth less or equal to **kinetics'** growth rate

Growth optimization heuristics:
[Feist and Palsson, 2010]

$$\begin{aligned}
\text{maximize} \quad & v_{\text{Growth}} \leq \text{Growth rate} \\
\text{such that:} \quad & S \cdot v = 0 \\
& l_r \leq v_r \leq u_r \quad \forall r \in \text{reactions} \\
& v_r = 0 \quad \forall r \in \text{inhibited reactions}
\end{aligned}$$

Regulatory state could not allow a higher growth rate than observed

General form of the compatibility constraints

$c(x) : \bigvee_i x_i \vee \bigvee_j \neg x_j$

$g(y)$: linear function

$$
\begin{aligned}
&\text{minimize} \quad f_{\text{obj}}(x) \\
&\text{such that} \\
&\quad \bigwedge_{\alpha} c_\alpha(x) \\
&\quad \wedge \quad \bigwedge_{\beta} c_\beta(x) \vee g_\beta(y) \leq 0 \\
&\quad \wedge \quad \forall z \in \mathbb{R}^p, \left(\bigwedge_{\gamma} c_\gamma(x) \vee g_\gamma(z) \leq 0 \right) \implies \left(\bigwedge_{\zeta} c_\zeta(x) \vee g_\zeta(z) \leq 0 \right) \\
&\text{with } x \in \mathbb{B}^n,\ y \in \mathbb{R}^m
\end{aligned}
$$

Criteria 1: regulatory state and **substrate state** are equal to **transcriptomics** and **kinetics**

Criteria 2: reaction activity states in the **metabolic state** and **fluxomics** are identical + **metabolic state** has the same growth rate as **kinetics**

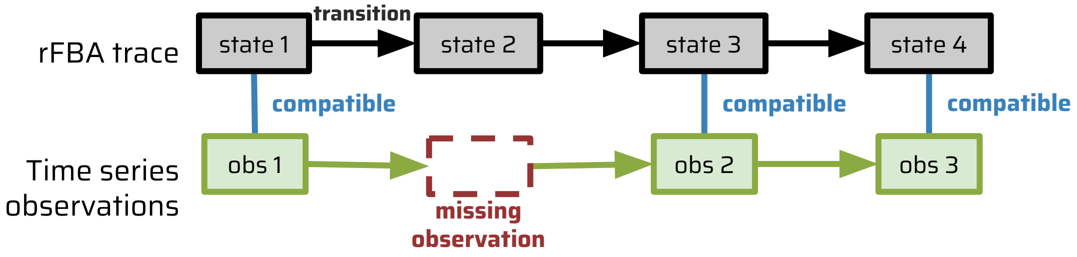
Criteria 3: all compatible **metabolic states** have a maximum growth less or equal to **kinetics**' growth rate

Compatibility criteria: logic (1) + linear (2) + quantified linear (3) constraints

Compatible Boolean networks

Compatible Boolean networks:

- ➔ Is **in the search space** described by the input interaction graph
- ➔ Has a **rFBA trace compatible with the time series observations**



- ➔ Is **optimal** according to:
  1. **Best fitting:** rFBA traces of minimal length compatible with time series
  2. **Parsimony:** subset minimal Boolean networks

General form of the inference problem

$c(x) : \bigvee_i x_i \vee \bigvee_j \neg x_j$

$g(y)$: linear function

$$\begin{aligned} &\text{minimize} \quad f_{\text{obj}}(x) \\ &\text{such that} \end{aligned}$$

Optimization criteria

$$\bigwedge_{\alpha} c_{\alpha}(x)$$

Search space + Regulatory state + Compatibility - criteria 1

$$\wedge \quad \bigwedge_{\beta} c_{\beta}(x) \vee g_{\beta}(y) \leq 0$$

Metabolic state + Compatibility - criteria 2

$$\wedge \quad \forall z \in \mathbb{R}^p, \left(\bigwedge_{\gamma} c_{\gamma}(x) \vee g_{\gamma}(z) \leq 0 \right) \implies \left(\bigwedge_{\zeta} c_{\zeta}(x) \vee g_{\zeta}(z) \leq 0 \right)$$

Compatibility - criteria 3

with $x \in \mathbb{B}^n$, $y \in \mathbb{R}^m$

Problem form:

Combinatorial optimization problem modulo quantified linear constraints

Contributions: two formulations of the inference problem

Two quantified formulations of the inference problem:

1. **Boolean over-approximation**
   - Boolean satisfiability problem with 2 levels of quantifiers (2-QBF)
   - Based on our own Boolean over-approximation of the rFBA dynamics
   - **Publication:** *Learning Boolean Controls in Regulated Metabolic Networks: A Case-Study.* **CMSB** *2021.*

2. **Flux-based formulation**
   - Combinatorial optimization problem modulo quantified linear constraints (OPT+qLP)
   - **Publications:**

     Bioinformatics: *MERRIN: MEtabolic Regulation Rule INference from time series data.* **Bioinformatics** *2022.*

     Formal methods: *CEGAR-Based Approach for Solving Combinatorial Optimization Modulo Quantified Linear Arithmetics Problems.* **AAAI** *2024.*

Contributions: two formulations of the inference problem

Two quantified formulations of the inference problem:

1. **Boolean over-approximation**
   - Boolean satisfiability problem with 2 levels of quantifiers (2-QBF)
   - Based on our own Boolean over-approximation of the rFBA dynamics
   - **Publication:** *Learning Boolean Controls in Regulated Metabolic Networks: A Case-Study.* **CMSB** *2021.*

2. **Flux-based formulation**
   - Combinatorial optimization problem modulo quantified linear constraints (OPT+qLP)
   - **Publications:**

     Bioinformatics: *MERRIN: MEtabolic Regulation Rule INference from time series data.* **Bioinformatics** *2022.*
     Formal methods: *CEGAR-Based Approach for Solving Combinatorial Optimization Modulo Quantified Linear Arithmetics Problems.* **AAAI** *2024.*

In this presentation

Contribution 2:

Solving

Combinatorial optimization problem modulo quantified linear constraints - OPT+qLP

$c(x)$: $\bigvee_i x_i \vee \bigvee_j \neg x_j$

$g(y)$: linear function

$$
\begin{aligned}
&\text{minimize} \quad f_{\text{obj}}(x) \\
&\text{such that} \\
&\bigwedge_\alpha c_\alpha(x) \\
&\wedge \quad \bigwedge_\beta c_\beta(x) \vee g_\beta(y) \leq 0 \\
&\wedge \quad \forall z \in \mathbb{R}^p, \left(\bigwedge_\gamma c_\gamma(x) \vee g_\gamma(z) \leq 0\right) \implies \left(\bigwedge_\zeta c_\zeta(x) \vee g_\zeta(z) \leq 0\right) \\
&\text{with } x \in \mathbb{B}^n,\ y \in \mathbb{R}^m
\end{aligned}
$$

How to enumerate solutions of an OPT+qLP problem?

Combinatorial optimization problem modulo quantified linear constraints - OPT+qLP

$c(x) : \bigvee_i x_i \vee \bigvee_j \neg x_j$

$g(y)$: linear function

$$\begin{aligned}
&\text{minimize} \quad f_{\text{obj}}(x) \\
&\text{such that} \\
&\bigwedge_\alpha c_\alpha(x) \\
&\wedge \quad \bigwedge_\beta c_\beta(x) \vee g_\beta(y) \leq 0 \\
&\wedge \quad \forall z \in \mathbb{R}^p, \left(\bigwedge_\gamma c_\gamma(x) \vee g_\gamma(z) \leq 0 \right) \implies \left(\bigwedge_\zeta c_\zeta(x) \vee g_\zeta(z) \leq 0 \right) \\
&\text{with } x \in \mathbb{B}^n,\ y \in \mathbb{R}^m
\end{aligned}$$

OPT problem

MaxSAT,
ASP

How to enumerate solutions of an OPT+qLP problem?

Combinatorial optimization problem modulo quantified linear constraints - OPT+qLP

$c(x) : \bigvee_i x_i \vee \bigvee_j \neg x_j$

$g(y)$: linear function

$$\text{minimize} \quad f_{\text{obj}}(x)$$

$$\text{such that}$$

$$\bigwedge_\alpha c_\alpha(x)$$

OPT problem

MaxSAT,
ASP

$$\wedge \quad \bigwedge_\beta c_\beta(x) \vee g_\beta(y) \leq 0$$

OPT problem with linear constraints

Conflict driven clause learning (CDCL) methods [Marques-Silva and Sakallah, 1996]

SMT solvers (e.g. z3),
ASP modulo theory (e.g. *Clingo[lpx]*)

$$\wedge \quad \forall z \in \mathbb{R}^p, \left(\bigwedge_\gamma c_\gamma(x) \vee g_\gamma(z) \leq 0 \right) \implies \left(\bigwedge_\zeta c_\zeta(x) \vee g_\zeta(z) \leq 0 \right)$$

$$\text{with } x \in \mathbb{B}^n,\ y \in \mathbb{R}^m$$

How to enumerate solutions of an OPT+qLP problem?

Combinatorial optimization problem modulo quantified linear constraints - OPT+qLP

$c(x) : \bigvee_i x_i \vee \bigvee_j \neg x_j$

$g(y)$: linear function

$$
\begin{aligned}
&\text{minimize} \quad f_{\text{obj}}(x) \\
&\text{such that} \\
&\bigwedge_{\alpha} c_\alpha(x) \\
\wedge \quad &\bigwedge_{\beta} c_\beta(x) \vee g_\beta(y) \leq 0 \\
\wedge \quad &\forall z \in \mathbb{R}^p, \left(\bigwedge_{\gamma} c_\gamma(x) \vee g_\gamma(z) \leq 0 \right) \implies \left(\bigwedge_{\zeta} c_\zeta(x) \vee g_\zeta(z) \leq 0 \right) \\
&\text{with } x \in \mathbb{B}^n,\ y \in \mathbb{R}^m
\end{aligned}
$$

OPT problem

MaxSAT, ASP

OPT problem with linear constraints

Conflict driven clause learning (CDCL) methods [Marques-Silva and Sakallah, 1996]

SMT solvers (e.g. z3),
ASP modulo theory (e.g. *Clingo[lpx]*)

OPT+qLP: OPT problem with one level of quantified linear constraints

Methods mainly rely on:

1. *E-matching* [De Moura and Bjørner, 2007]
2. *Quantifier elimination*
3. *CDCL-based methods*

No solver natively supports linear quantifiers, optimization, and enumeration

Contributions’ outlines

Two methods to address OPT+qLP problems:

1. **Constraint learning**
   Rely on structural property of OPT+qLP problems

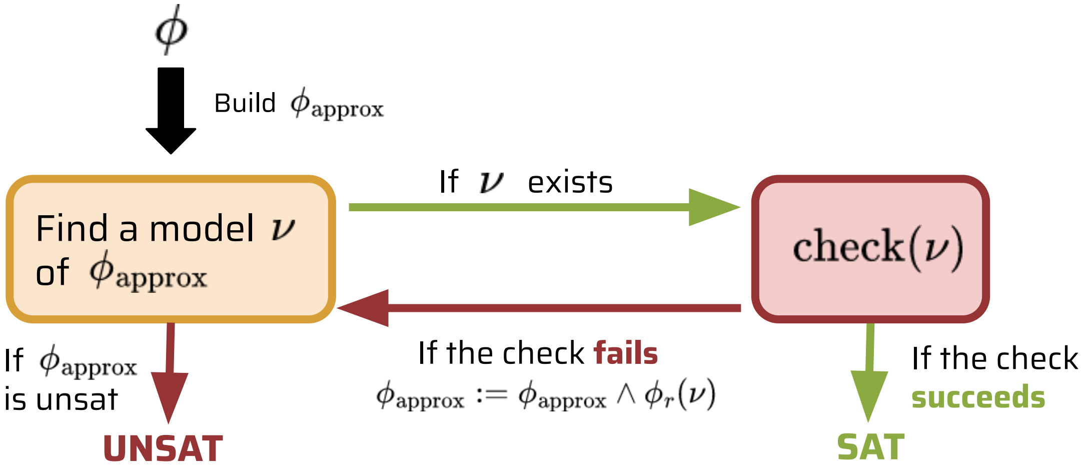
2. **Universal quantifier elimination**
   Remove universal quantifiers
   Usable with state-of-the-art OPT+LP solvers
   e.g. clingo[lpx] [Janhunen et al. 2017], *z3* [De Moura and Bjørner 2008]

Counter-Example Guided Abstraction Refinement – CEGAR[1]

Rely on:

1. An **over-approximation** of the OPT+qLP problem $\phi \implies \phi_{\text{approx}}$
2. Methods to **check** the validity of an assignment $\text{check}(\nu)$
3. **Refinement functions** to generalize counter-examples $\phi_r(\nu)$

Workflow:


ϕ
Build ϕ_{approx}
Find a model ν of ϕ_{approx}
If ν exists
$\text{check}(\nu)$
If the check **fails**
$\phi_{\text{approx}} := \phi_{\text{approx}} \wedge \phi_r(\nu)$
If ϕ_{approx} is unsat
UNSAT
If the check **succeeds**
SAT

Conflict Driven Constraint Learning (*CDCL*)-like solving framework

[1] E. Clarke *et al.*, **Journal of the ACM**, 2003

Boolean over-approximation

Replace linear constraints by Boolean variables

True: *linear constraint must hold*

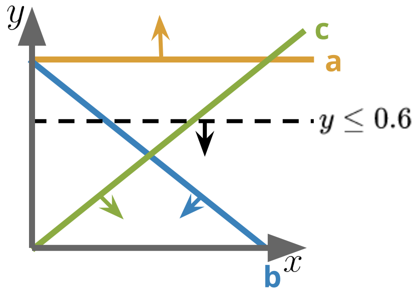
False: *ignored the linear constraint*

— Proof in [Thuillier *et al.*, 2024]

$$\frac{\begin{array}{l} \bigwedge_{c \in C} c(x) \\ \wedge \quad \bigwedge_{d \in D} d(x,y) \\ \wedge \quad \forall z \in \mathbb{R}^p, \bigwedge_{e \in E} e(x,z) \implies \bigwedge_{h \in H} h(x,z) \end{array}}{\phi}$$

Example:

$$\begin{array}{l} \text{minimize} \quad a+b+c \\ \text{such that} \\ \quad (a \vee b \vee c) \\ \quad \wedge \forall x,y \in \mathbb{R}, \begin{pmatrix} & (y \geq 1 \vee \neg a) \\ \wedge & (x+y \leq 1 \vee \neg b) \\ \wedge & (-x+y \leq 0 \vee \neg c) \end{pmatrix} \implies y \leq 0.6 \\ \text{with} \;\; a,b,c \in \mathbb{B} \end{array}$$



Linear search space

OPT+qLP problems can be approximated by Boolean optimization problems

Boolean over-approximation

Replace linear constraints by Boolean variables

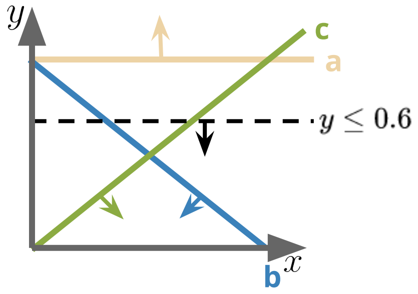
- **True:** *linear constraint must hold*
- **False:** *ignored the linear constraint*

— Proof in [Thuillier *et al.*, 2024]

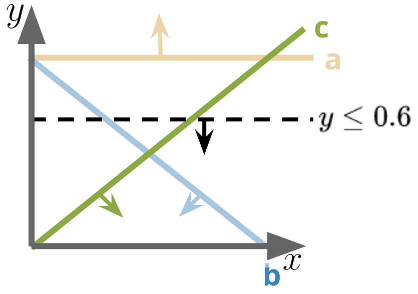
$$\underbrace{\bigwedge_{c \in C} c(x) \land \bigwedge_{d \in D} d(x,y) \land \boxed{\forall z \in \mathbb{R}^p}, \bigwedge_{e \in E} e(x,z) \boxed{\implies} \bigwedge_{h \in H} h(x,z)}_{\phi} \implies \underbrace{\bigwedge_{c \in C} c(x) \land \bigwedge_{d \in D} \bar{d}(x,\bar{f}_d) \land \bigwedge_{e \in E} \bar{e}(x,\bar{f}_e) \boxed{\land} \bigwedge_{h \in H} \bar{h}(x,\bar{f}_h)}_{\phi_{\text{approx}}}$$

Example:

$$\begin{aligned}
&\text{minimize} \quad a + b + c \\
&\text{such that} \\
&\quad (a \lor b \lor c) \\
&\quad \land \cancel{\boxed{\forall x, y \in \mathbb{R},}} \left(\begin{array}{ll} & (y \geq 1 \lor \neg a) \\ \land & (x + y \leq 1 \lor \neg b) \\ \land & (-x + y \leq 0 \lor \neg c) \end{array} \right) \overset{\land}{\cancel{\boxed{\implies}}} y \leq 0.6 \\
&\text{with} \;\; a, b, c \in \mathbb{B}
\end{aligned}$$



Linear search space for $\{b, c\}$



Linear search space for $\{c\}$

OPT+qLP problems can be approximated by Boolean optimization problems

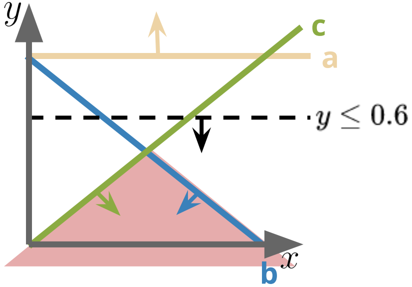
Checking quantified linear constraints

Given set of quantified linear constraints that must hold:

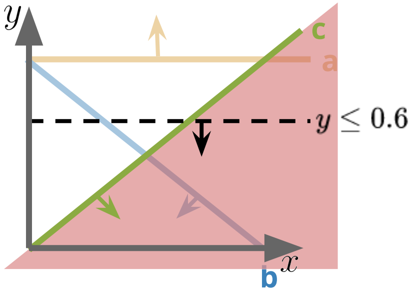
$$\forall y \in \mathbb{R}^p,\ \bigwedge_f f(y) \leq 0 \implies g(y) \leq 0 \quad \iff \quad \begin{array}{rll} \text{maximize} & g(y) & \leq 0 \\ \text{such that:} & f(y) \leq 0 & \forall f \\ \text{with} & y \in \mathbb{R}^p & \end{array}$$

Example:

$$\begin{array}{l} \text{minimize} \quad a + b + c \\ \text{such that} \\ \quad (a \lor b \lor c) \\ \quad \land \boxed{\cancel{\forall x, y \in \mathbb{R},}} \left(\begin{array}{rc} & (y \geq 1 \lor \neg a) \\ \land & (x + y \leq 1 \lor \neg b) \\ \land & (-x + y \leq 0 \lor \neg c) \end{array} \right) \overset{\land}{\boxed{\cancel{\implies}}} y \leq 0.6 \\ \text{with} \ \ a, b, c \in \mathbb{B} \end{array}$$



Linear search space for $\{b, c\}$



Linear search space for $\{c\}$

Checking quantified linear constraints = solving a linear optimization problem

Checking quantified linear constraints

Given set of quantified linear constraints that must hold:

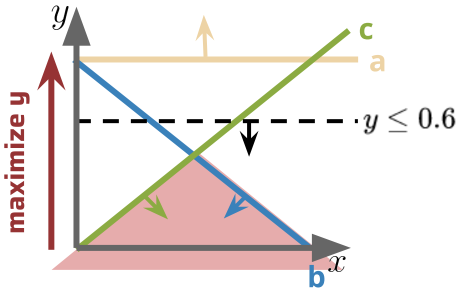
$$\forall y \in \mathbb{R}^p, \bigwedge_f f(y) \leq 0 \implies g(y) \leq 0 \iff \begin{array}{rll} \text{maximize} & g(y) & \leq 0 \\ \text{such that:} & f(y) \leq 0 & \forall f \\ \text{with} & y \in \mathbb{R}^p & \end{array}$$

Example:

$$\begin{array}{l} \text{minimize} \quad a + b + c \\ \text{such that} \\ \quad (a \vee b \vee c) \\ \quad \wedge \boxed{\cancel{\forall x, y \in \mathbb{R},}} \left(\begin{array}{rc} & (y \geq 1 \vee \neg a) \\ \wedge & (x + y \leq 1 \vee \neg b) \\ \wedge & (-x + y \leq 0 \vee \neg c) \end{array} \right) \overset{\wedge}{\boxed{\cancel{\implies}}} y \leq 0.6 \\ \text{with} \;\; a, b, c \in \mathbb{B} \end{array}$$

Valid assignment

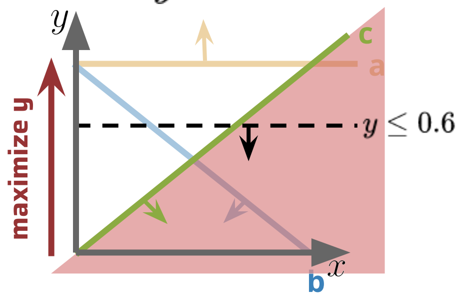
$\max \;\; y = 0.5$



Linear search space
for $\{b, c\}$

Invalid assignment

$\max \;\; y = \infty$



Linear search space
for $\{c\}$

Checking quantified linear constraints = solving a linear optimization problem

Counter-examples generalization

Monotone property:

Adding linear constraints to a linear optimization problem could not increase its maximum

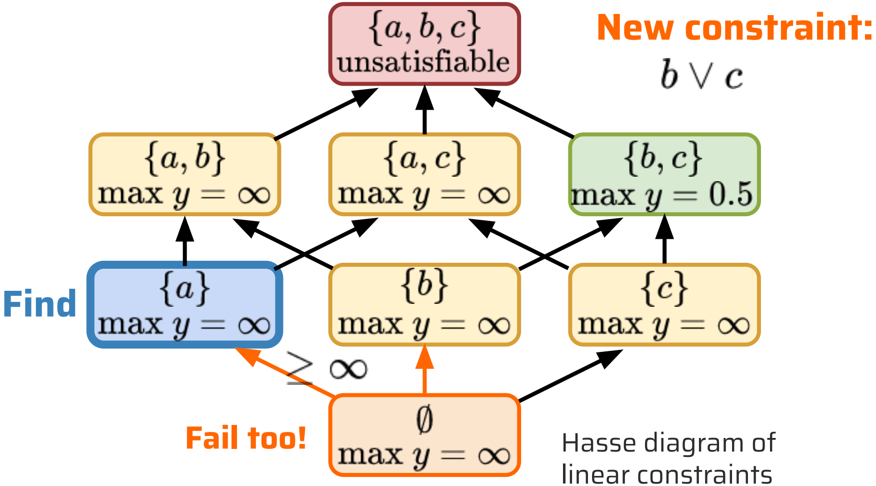
Example:

Objective: $\text{maximize} \quad y$

Constraints

$$\begin{aligned} a &: y \geq 1 \\ b &: x + y \leq 1 \\ c &: -x + y \leq 0 \end{aligned}$$

Variables: $x, y \in \mathbb{R}$



Hasse diagram of linear constraints power set

Set of linear constraints fails the check $\implies$ all its subset will fail too

Further refinements

Optimal core:
Largest superset of linear constraints having a maximum failing the linear check

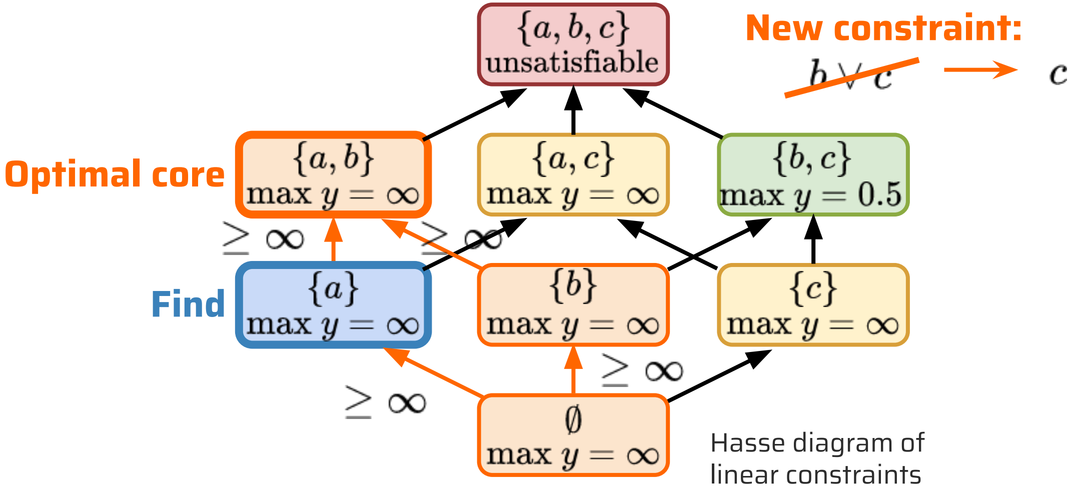
Example:

Objective: maximize y

Constraints

$$\begin{aligned} a &: y \geq 1 \\ b &: x + y \leq 1 \\ c &: -x + y \leq 0 \end{aligned}$$

Variables: $x, y \in \mathbb{R}$



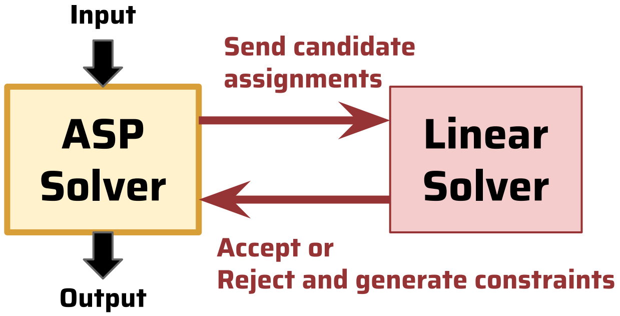
Hasse diagram of linear constraints power set

Similar to unsatisfiable cores but applied to optimum values

Implementations

Based on Answer Set Programming (ASP):
Logic programming
Handle optimization and efficient enumeration [Gebser *et al.*, 2011-13]

Linear checks made with generic linear solvers
CBC, GLPK, Gurobi



Tools:

- ➔ **MERRIN:** inference of Boolean network controlling metabolic networks
  Published in Bioinformatics / at ECCB22 [Thuillier *et al.*, 2022] *github.com/bioasp/merrin*
- ➔ **MerrinASP:** generic solver for OPT+qLP problems
  Published at AAAI24 [Thuillier *et al.*, 2024] *github.com/kthuillier/merrinasp*

Two implementations of the CEGAR-based workflow based on ASP

Alternative method: quantifier elimination

From **weak duality theorem:**

Universally quantified constraint ***Quantifier-free constraint***

$$\forall z \in \mathbb{R}^p, A \cdot z \leq b \implies c^T \cdot z \leq \lambda \quad \xrightarrow[\text{satisfiable}]{\text{If } A \cdot z \leq b} \quad \exists y \in \mathbb{R}^{q+}, A^T \cdot y = c \wedge b^T \cdot \leq \lambda$$

Inconvenient: only an over-approximation in general case

Advantage: usable with any quantifier-free SAT+LP solvers *e.g. clingo[lpx] or z3*

Benchmarked on the inference problem

- No automated rewriting implementation → manual rewriting
- 10 times slower than CEGAR-based method

Quantifier elimination rewriting to solve OPT+qLP problems with any SAT+LP solver

Contribution 3:

Benchmarking

Escherichia coli models

Core-carbon metabolism [Covert *et al.*, 2001]

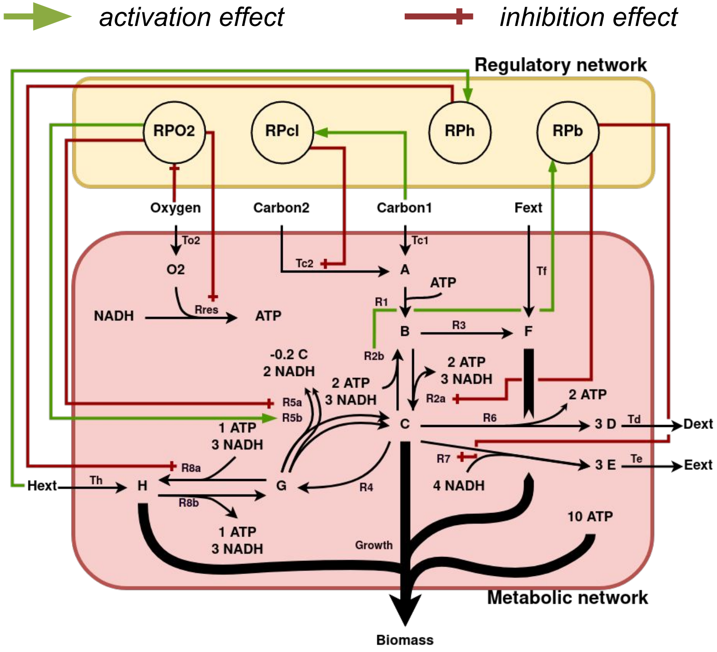
Core-carbon model

- *20 reactions*
- *11 regulatory rules*

***E. coli* core-metabolism [Covert *et al.*, 2002]**

Medium-scale model

- *113 reactions*
- 151 regulatory rules



Core-carbon metabolism from [Covert *et al.*, 2001]

Time series generation protocol [Thuillier *et al.*, 2022]

From rFBA simulations → noisy time series with different data types

Generation of two synthetic datasets of increasing size based on *e. coli* models

Generated benchmarks

	Instances	Type combinations	Noise range	Repetition	Number of variables		Number of constraints	
					Boolean	Linear	Logic	Linear
Core-carbon	240	F, K, T K, T F T	0% - 50%	10	6.5×10^4	4×10^3	2.7×10^5	1.2×10^4
Medium-scale	60	F, K, T	0%	60	$\mathbf{4 \times 10^9}$	$\mathbf{16 \times 10^3}$	$\mathbf{18 \times 10^5}$	$\mathbf{5 \times 10^4}$

$x10^5$

x4

Core-carbon benchmark:

Impact of observation types and noise rates on the inference

Medium-scale benchmark:

Impact of model size – ensure scalability

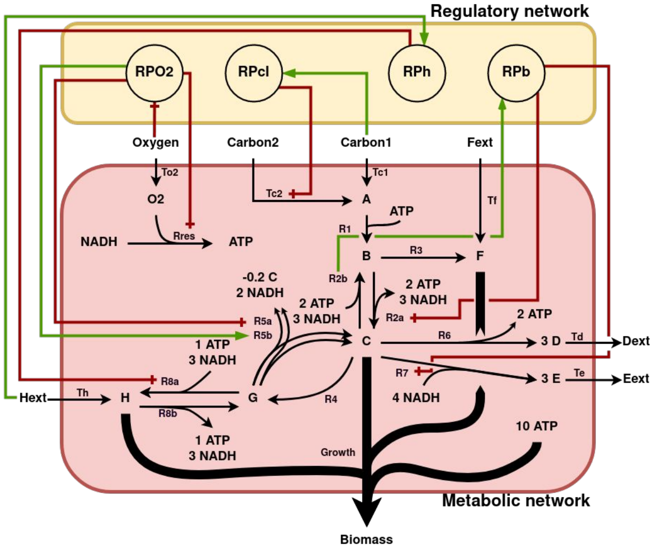
Application on core-carbon model

Complete data

Data types: *Fluxomics, Kinetics, Transcriptomics*
Noise: *0%*

48 Boolean networks
Gold standard is inferred

Computation time: 7s



Gold standard from [Covert *et al.*, 2001]

Gold standard network is inferred from complete observations

Application on core-carbon model

Complete data

Data types: *Fluxomics, Kinetics, Transcriptomics*
Noise: *0%*

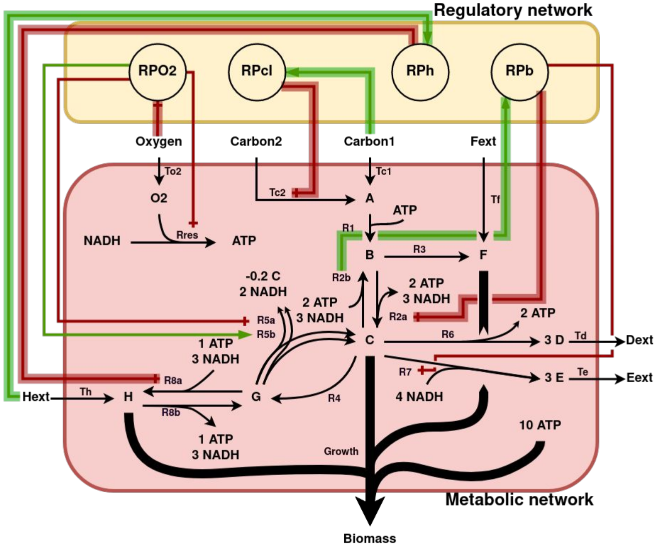
48 Boolean networks

Gold standard is inferred

Computation time: 7s

1 subset minimal Boolean network

- Reproduce exactly the input rFBA time series
  Residual Sum of Squares (RSS): 0
- Smaller than gold standard
  Precision: 1 / Recall: 0.64
  - Not all regulatory rules are retrieved
  - Consistent with [Covert *et al.*, 2001]



Subset minimal model

rFBA formalism does not allow capturing all regulatory process

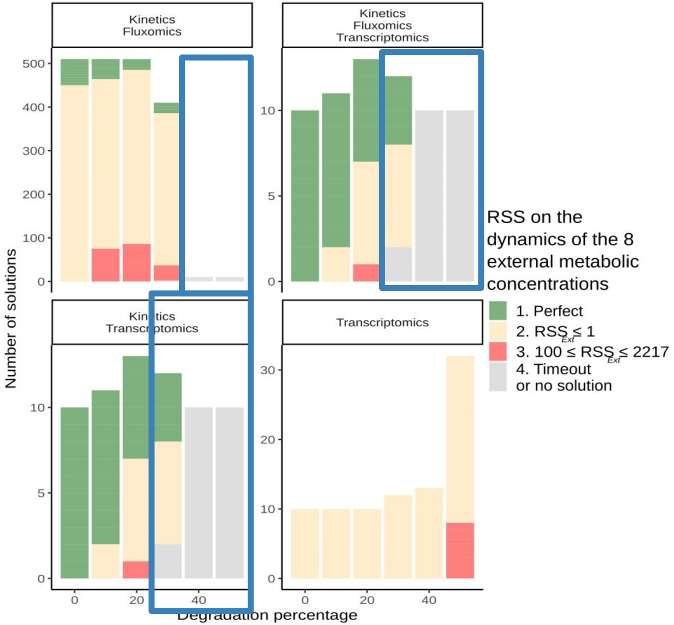
Perspective: **upgrade modeling formalisms to capture missing rules**

Impact of noise and data types

Benchmark

Data types: *4 combinations*
Noise: *0% - 50%*

Compute subset minimal models for each instance
Unsatisfiable instance due to noise in fluxomics and kinetics



Impact of types combination on *RSS*

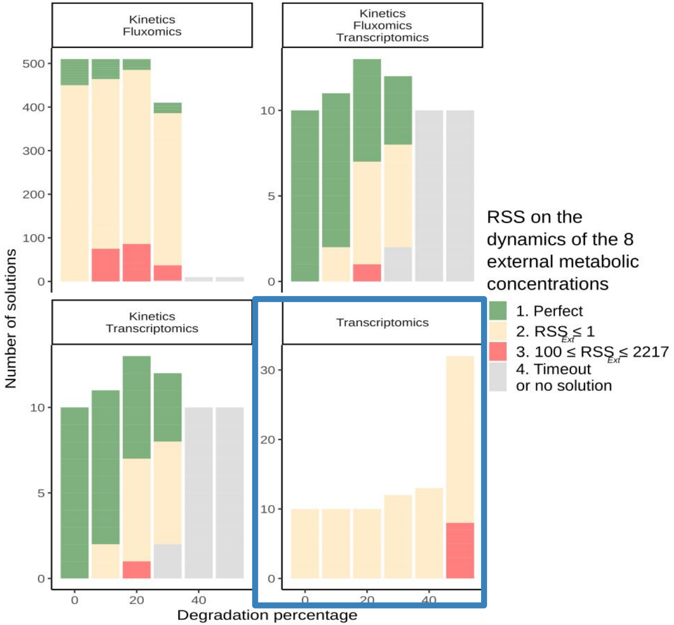
MERRIN handles up to 20% of noise

Impact of noise and data types

Benchmark

Data types: *4 combinations*
Noise: *0% - 50%*

Compute subset minimal models for each instance
Unsatisfiable instance due to noise in fluxomics and kinetics



Impact of types combination on *RSS*

Transcriptomics only: 1 control rule is never inferred

Metabolic observations are needed to exactly reproduce input rFBA time series

Impact of noise and data types

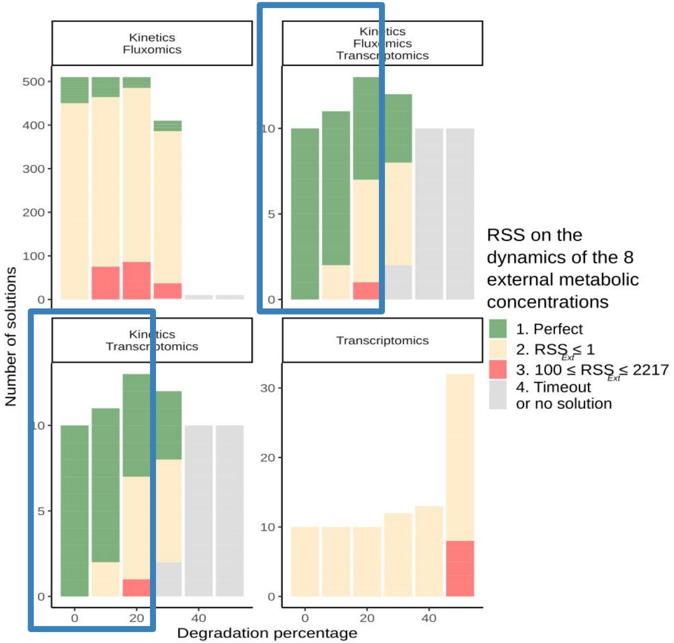
Benchmark

Data types: *4 combinations*
Noise: *0% - 50%*

Compute subset minimal models for each instance
Unsatisfiable instance due to noise in fluxomics and kinetics

Fluxomics is not necessary if there is kinetics
Same results for [K,F,T] and [K,F]

Type: transcriptomics + kinetics / noise: 0% - 20%
- ➔ *RSS < 1*
- ➔ *Precision: 1 / Recall: 0.64*



Impact of types combination on *RSS*

Transcriptomics + kinetics: sufficient to infer regulations controlling the metabolism

Handle up to 20% of noise

Application on *E. coli* core-metabolism[1]

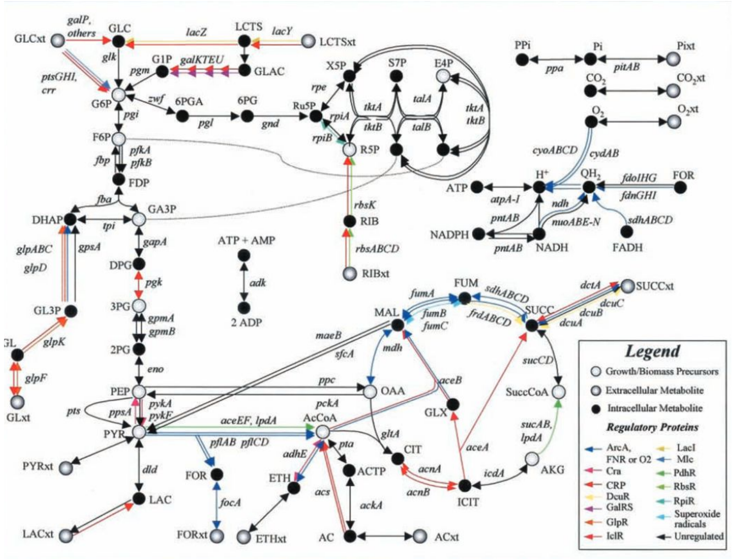
Complete data - 3 time series

Data types: *Fluxomics, Kinetics, Transcriptomics*
Noise: *0%*

838 860 800 subset minimal Boolean networks

Computation time: < 8h

All subset minimal models are enumerated



E. coli core-metabolism from [Covert *et al.*, 2002]

MERRIN's method scales to medium-scale models

[1] M. W. Covert and B. Ø. Palsson, **Journal of biological chemistry**, 2002

Application on *E. coli* core-metabolism[1]

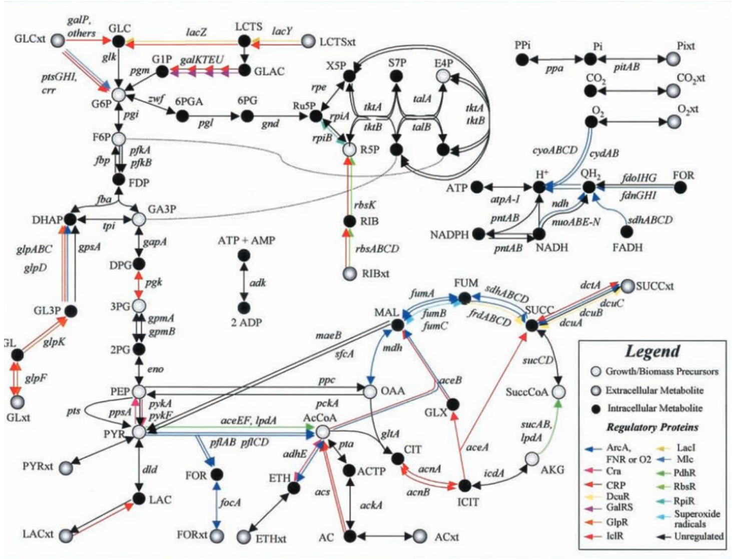
Complete data - 3 time series

Data types: *Fluxomics, Kinetics, Transcriptomics*
Noise: *0%*

838 860 800 subset minimal Boolean networks
Computation time: < 8h
All subset minimal models are enumerated

Compatibility with time series
Residual Sum of Squares: 0

Smaller than gold standard model
Precision: ~0.87 / Recall: ~0.11



E. coli core-metabolism from [Covert *et al.*, 2002]

Rules may not be inferred due to data incompleteness

[1] M. W. Covert and B. Ø. Palsson, **Journal of biological chemistry**, 2002

Conclusion and Perspectives

Conclusion - *general*

Thesis' question: can we infer regulatory controls of the metabolism?

Inference problem formalization:

→ No method to infer controls of the metabolism

→ Integrate both scale dynamics: **discrete** + **flux-based**

→ **Combinatorial optimization problem modulo quantified linear constraints (OPT+qLP)**

Solving OPT+qLP problems:

→ Existing hybrid solvers do not handle such problem

→ Developed our **own solving methods**: CEGAR-based + quantifier elimination

Benchmarking:

→ Generate synthetic datasets of 2 *E. coli* models

→ Study the impact of: noise, observations types, and instance size

Conclusion - *for bioinformatics*

Two formulations of the inference problem:

1. **Boolean relaxation** — *Boolean satisfiability with two levels of quantifiers (2-QBF)*
   Based on Boolean approximation of rFBA dynamics
   Paper: Learning Boolean Controls in Regulated Metabolic Networks: A Case-Study. **CMSB** 2021

2. **Hybrid** — *Combinatorial optimization problem modulo quantified linear constraints (OPT+qLP)*
   Based on rFBA dynamics
   Paper: MERRIN: MEtabolic Regulation Rule INference from time series data. **Bioinformatics** 2022.

MERRIN: dedicated implementation to solve the inference problem — *github.com/bioasp/merrin*
Support noisy kinetics, fluxomics, and transcriptomics observations
Benchmark on ***two synthetic datasets*** *based on* E. coli

Inference of control and feedback possible from kinetics and transcriptomics

Perspectives - *for bioinformatics*

Perspective 1: improving regulated metabolic networks simulation formalism

- ➔ rFBA framework is not sufficient to capture all regulatory rules
- ➔ Metabolic feedback sensors depends of specific concentration threshold

> **Use more precise simulation formalisms**
> *e.g.* r-deFBA [Liu and Bockmayr, 2020]

Perspective 2: updating Boolean networks controlling metabolic networks

- ➔ Some regulatory rules are already known and experimentally validated
- ➔ Inefficient to infer regulatory *de novo* each time new experiments are available

> **Develop methods to update Boolean networks to account for new observations**

Conclusion - *for formal methods*

Inference problem has specificities not handled by existing solvers
Optimality criteria, enumeration constraints, quantified hybrid constraints

Two methods to address OPT+qLP problems:

1. **CEGAR-based**
   Paper: CEGAR-Based Approach for Solving Combinatorial Optimization Modulo Quantified Linear Arithmetics Problems. **AAAI** 2024.

2. **Quantifier-elimination** — over-approximate quantified formulas
   Manually performed on the inference problem
   Usable with state-of-the-art hybrid solvers — *e.g. clingo[lpx]* [Janhunen *et al.*, 2017] *or z3* [De Moura and Bjørner, 2008]

MerrinASP: generic solver for OPT+qLP problems based on ASP — *github.com/kthuillier/merrinasp*
CEGAR-based extension of the ASP solver clingo [Gebser *et al.*, 2017]

CEGAR-based method is 10 time faster than quantifier-elimination

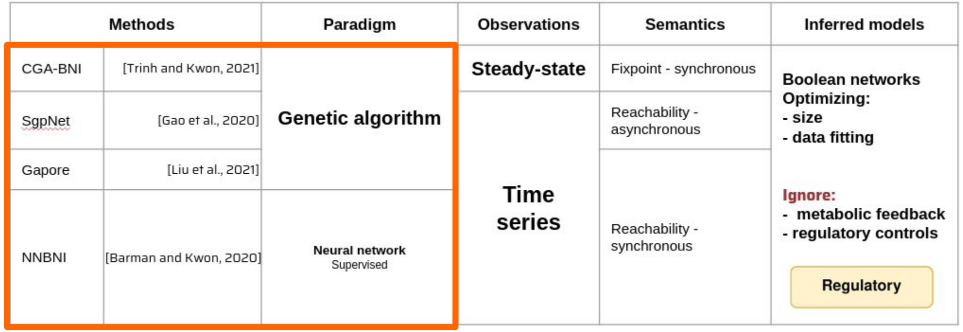
Perspective - *for formal methods*

Perspective 3: inferring missing interactions at runtime

➔ Not all interactions are known

➔ MERRIN’s results depend on the input interaction graph
Missing interaction leads to unsatisfiable solutions

➔ Statistical inference methods learn interactions at runtime



Methods		Paradigm	Observations	Semantics	Inferred models
CGA-BNI	[Trinh and Kwon, 2021]	Genetic algorithm	Steady-state	Fixpoint - synchronous	Boolean networks Optimizing: - size - data fitting Ignore: - metabolic feedback - regulatory controls Regulatory
SgpNet	[Gao et al., 2020]		Time series	Reachability - asynchronous	
Gapore	[Liu et al., 2021]			Reachability - synchronous	
NNBNI	[Barman and Kwon, 2020]	Neural network Supervised			

Iteratively refine the interaction graph and infer regulatory rules

Acknowledgements

Jury members:

Emmanuelle Becker
François Fages
Simon de Givry
Misbah Razzaq
Laurent Tournier

Supervisors:

Anne Siegel
Loïc Paulevé

CSI members:

Laurent Tournier
Charlotte Truchet

The whole Symbiose's teams!
(in particular *Dyliss*)

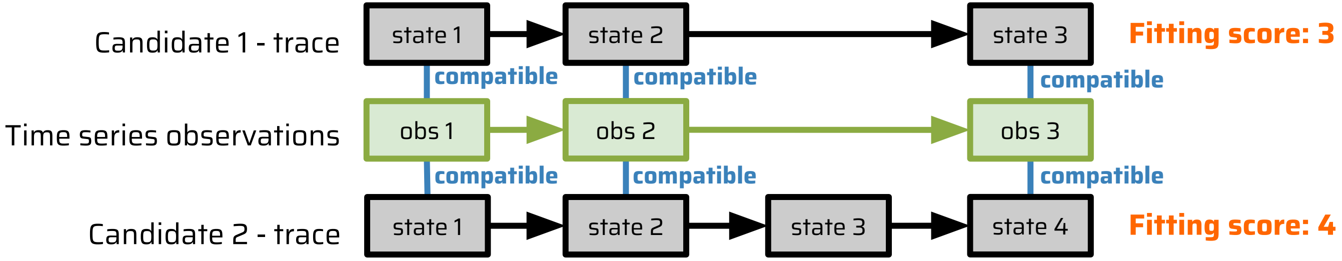
… and the cats!

Université de Rennes   UMR IRISA   cnrs   Inria   DyLiSS

Appendices

Optimization criteria

1. **Best fitting:** traces of minimal length compatible with observations



2. **Parsimony:** subset minimal Boolean networks

$$\begin{gathered} f_{\text{lacZ}}(x) = \neg x_{\text{GLCxt}} \\ f_{\text{galKTEU}}(x) = \neg x_{\text{GLCxt}} \end{gathered} \subset \begin{gathered} f_{\text{lacZ}}(x) = \neg x_{\text{GLCxt}} \\ f_{\text{galKTEU}}(x) = \neg x_{\text{GLCxt}} \wedge \neg x_{\text{GalR}} \end{gathered} \subset \begin{gathered} f_{\text{lacZ}}(x) = \neg x_{\text{GLCxt}} \\ f_{\text{galKTEU}}(x) = \neg x_{\text{GLCxt}} \wedge \neg x_{\text{GalR}} \\ f_{\text{GalR}}(x) = \neg x_{\text{LCTSxt}} \end{gathered}$$

subset minimal

Combinatorial optimization criteria: minimize trace length, minimize size

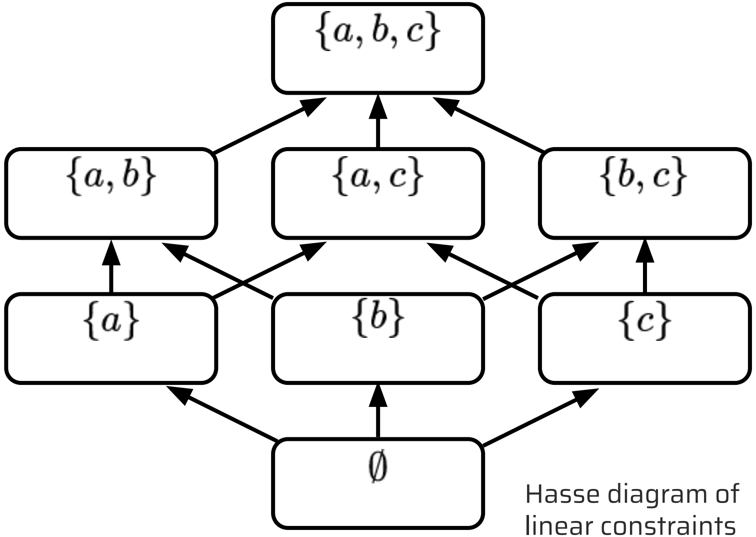
Our CEGAR-based workflow in practice

Boolean over-approximation

$$
\begin{array}{l}
(a \vee b \vee c) \\
\wedge \boxed{\cancel{\forall x, y \in \mathbb{R},}} \left(\begin{array}{lc} & (y \geq 1 \vee \neg a) \\ \wedge & (x + y \leq 1 \vee \neg b) \\ \wedge & (-x + y \leq 0 \vee \neg c) \end{array} \right) \overset{\wedge}{\boxed{\cancel{\Longrightarrow}}} y \leq 0.6
\end{array}
$$

with $a, b, c \in \mathbb{B}$

Checked Boolean assignments:



Hasse diagram of linear constraints power set

Our CEGAR-based workflow in practice

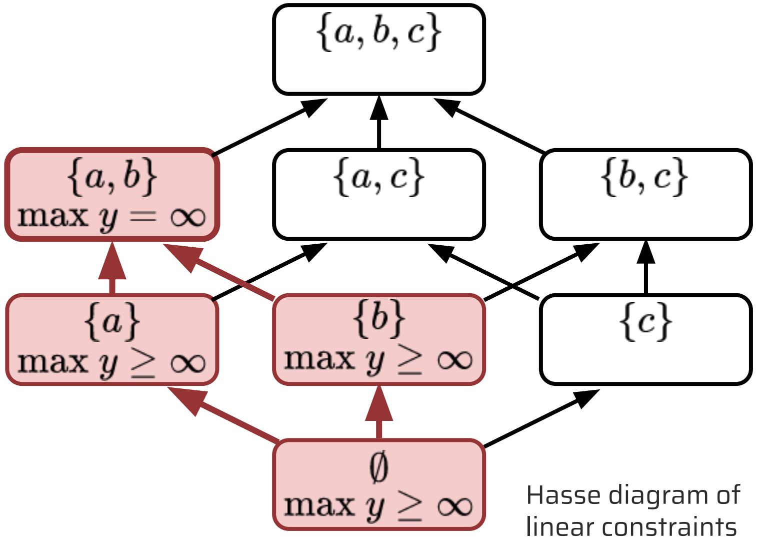
Boolean over-approximation

$$(a \lor b \lor c) \land \cancel{\forall x, y \in \mathbb{R},} \left(\begin{array}{rc} & (y \geq 1 \lor \neg a) \\ \land & (x + y \leq 1 \lor \neg b) \\ \land & (-x + y \leq 0 \lor \neg c) \end{array} \right) \overset{\land}{\cancel{\implies}} y \leq 0.6$$

with $a, b, c \in \mathbb{B}$

Checked Boolean assignments:

1. $\{a, b\} \longrightarrow \max y = \infty$


$\{a, b, c\}$
$\{a, b\}$ $\max y = \infty$
$\{a, c\}$
$\{b, c\}$
$\{a\}$ $\max y \geq \infty$
$\{b\}$ $\max y \geq \infty$
$\{c\}$
$\emptyset$ $\max y \geq \infty$

Hasse diagram of linear constraints power set

All subset of $\{a, b\}$ **have** $\max y \geq \infty$

Our CEGAR-based workflow in practice

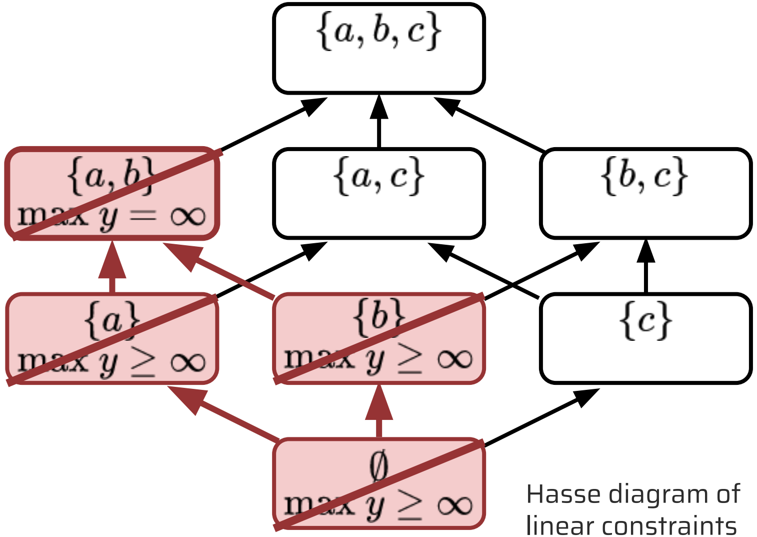
Boolean over-approximation

$$(a \lor b \lor c) \land (-x + y \le 0) \land \cancel{\forall x, y \in \mathbb{R},} \begin{pmatrix} & (y \ge 1 \lor \neg a) \\ \land & (x + y \le 1 \lor \neg b) \\ \land & (-x + y \le 0 \lor \neg c) \end{pmatrix} \overset{\land}{\cancel{\implies}} y \le 0.6$$

with $a, b, c \in \mathbb{B}$

Checked Boolean assignments:

1. $\{a, b\} \longrightarrow \max y = \infty$



Hasse diagram of linear constraints power set

All subset of $\{a, b\}$ **have** $\max y \ge \infty \Longrightarrow$ **Prohibit solutions without** $c : -x + y \le 0$

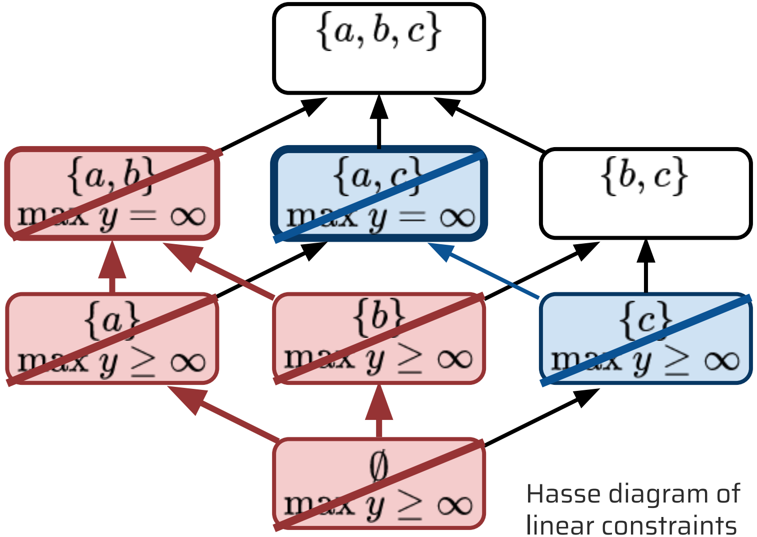
Our CEGAR-based workflow in practice

Boolean over-approximation

$$(a \lor b \lor c) \land (-x + y \le 0) \land (x + y \le 1)$$

$$\land \cancel{\forall x, y \in \mathbb{R},} \begin{pmatrix} & (y \ge 1 \lor \neg a) \\ \land & (x + y \le 1 \lor \neg b) \\ \land & (-x + y \le 0 \lor \neg c) \end{pmatrix} \overset{\land}{\cancel{\implies}} y \le 0.6$$

with $a, b, c \in \mathbb{B}$


$\{a, b, c\}$
$\{a, b\}$ $\cancel{\max y = \infty}$
$\{a, c\}$ $\cancel{\max y = \infty}$
$\{b, c\}$
$\{a\}$ $\cancel{\max y \ge \infty}$
$\{b\}$ $\cancel{\max y \ge \infty}$
$\{c\}$ $\cancel{\max y \ge \infty}$
$\emptyset$ $\cancel{\max y \ge \infty}$

Hasse diagram of linear constraints power set

Test Boolean assignments:

1. $\{a, b\} \longrightarrow \max y = \infty$
2. $\{a, c\} \longrightarrow \max y = \infty$

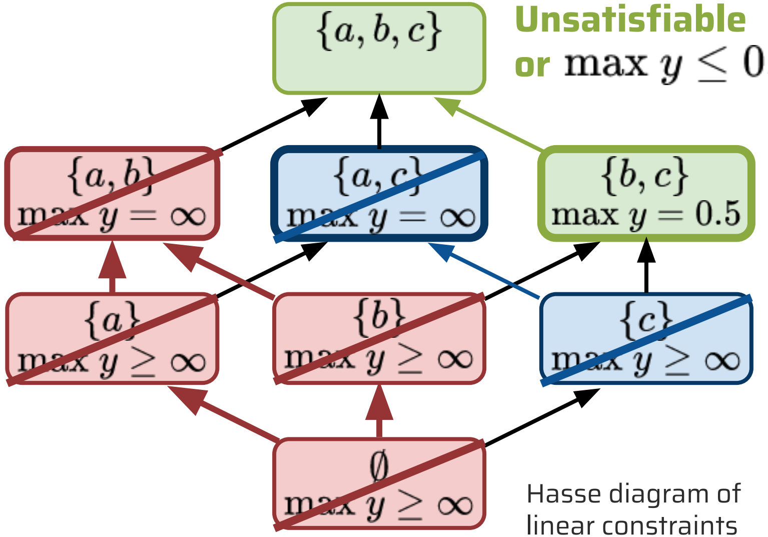
All subset of $\{a, b\}$ **have** $\max y \ge \infty \implies$ **Prohibit solutions without** $b : x + y \le 1$

Our CEGAR-based workflow in practice

Boolean over-approximation

$$(a \lor b \lor c) \land (-x + y \leq 0) \land (x + y \leq 1) \land \cancel{\forall x, y \in \mathbb{R},} \begin{pmatrix} (y \geq 1 \lor \neg a) \\ \land \quad (x + y \leq 1 \lor \neg b) \\ \land \quad (-x + y \leq 0 \lor \neg c) \end{pmatrix} \overset{\land}{\cancel{\implies}} y \leq 0.6$$

with $a, b, c \in \mathbb{B}$


$\{a, b, c\}$
$\{a, b\}$ max $y = \infty$
$\{a, c\}$ max $y = \infty$
$\{b, c\}$ max $y = 0.5$
$\{a\}$ max $y \geq \infty$
$\{b\}$ max $y \geq \infty$
$\{c\}$ max $y \geq \infty$
$\emptyset$ max $y \geq \infty$

Unsatisfiable or $\max y \leq 0.5$

Hasse diagram of linear constraints power set

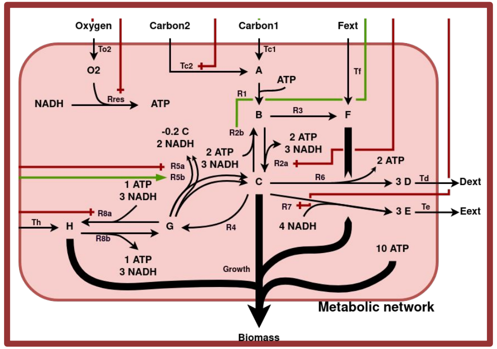
Test Boolean assignments:

1. $\{a, b\} \longrightarrow \max y = \infty$
2. $\{a, c\} \longrightarrow \max y = \infty$
3. $\{b, c\} \longrightarrow \max y = 0.5$

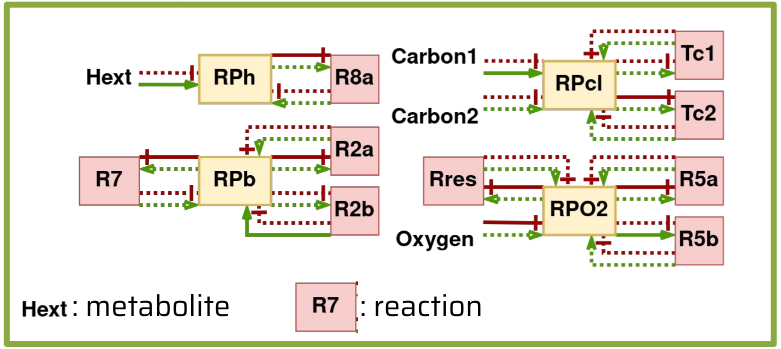
> **A valid assignment is found** $\implies$ **All supersets will be valid too**

Instance generation: application to core-carbon model

Metabolic network



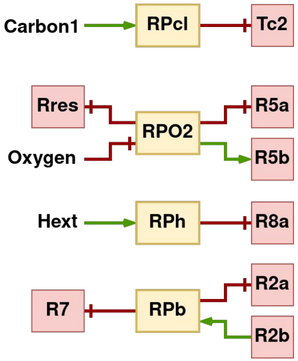
Interaction graph



Observations

Interaction graph

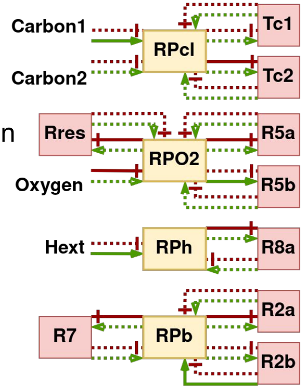
- ➔ Generate from model's regulatory networks
- ➔ Add noise: remove interaction signs and directions



Gold standard

➔ Remove interactions direction
➔ Remove interactions sign

➔ Add hypothetical regulation



Interaction graph

Time series generation workflow

Extract data per observation' types:

→ **Kinetics:** substrate concentrations

→ **Fluxomics:** reactions fluxes values

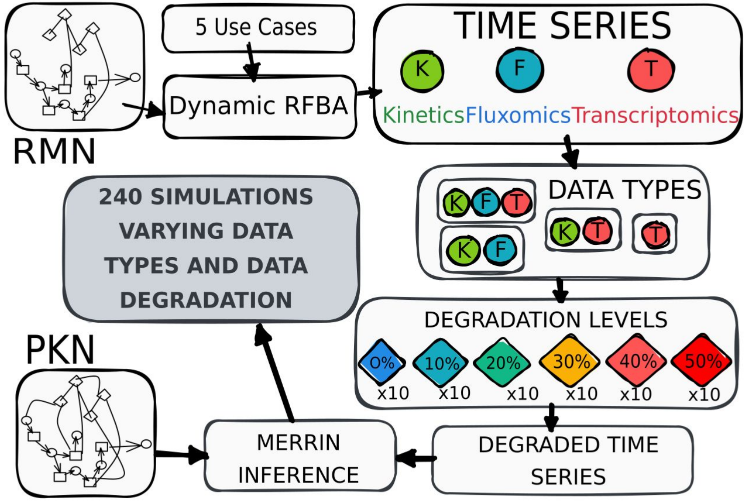
→ **Transcriptomics:** binarized observation

Keep simulation timesteps:

- 2 observations per growth phase
- 1 observation per transition

Noise:

- Noise on kinetics and fluxomics values
- Probability to remove an observed values
- Probability to remove a timestep

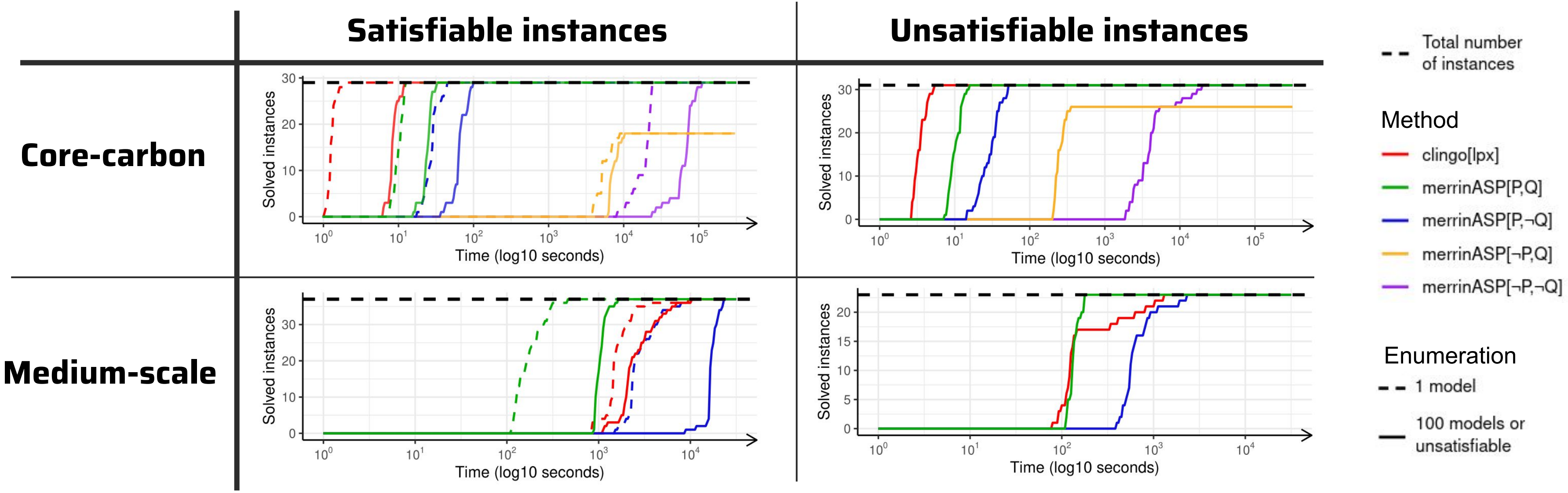


From [Thuillier *et al.*, 2022]

Generate noisy kinetics, fluxomics, and transcriptomics observations from rFBA simulations

MerrinASP - Performance comparison

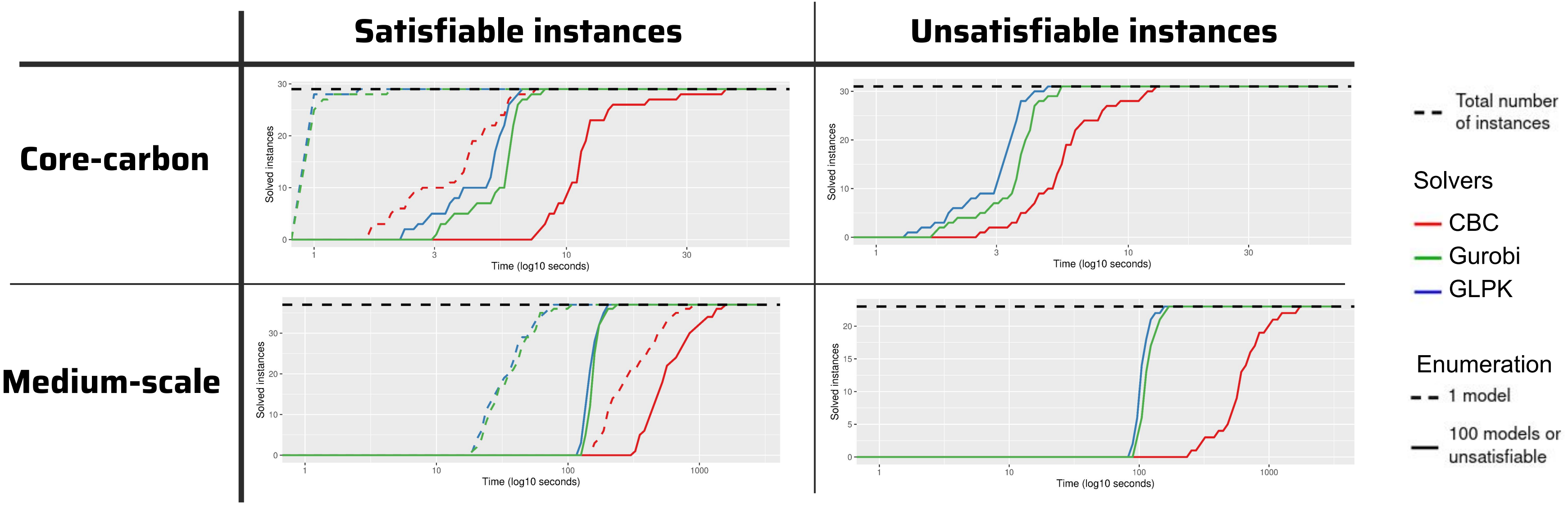
60 instances of the inference problem: transcriptomics, kinetics, and fluxomics
noise from 0% to 50%



CEGAR-based + partition: ~10 times faster than Clingo[lpx] + quantifier elimination

Impact of linear solvers

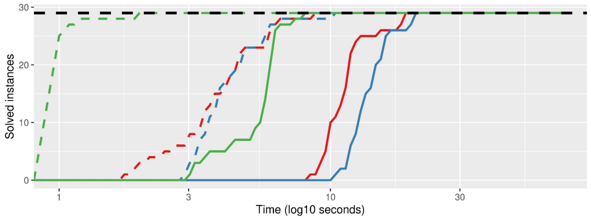
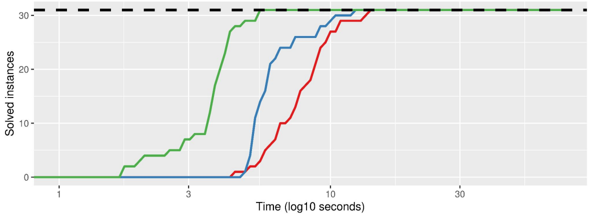
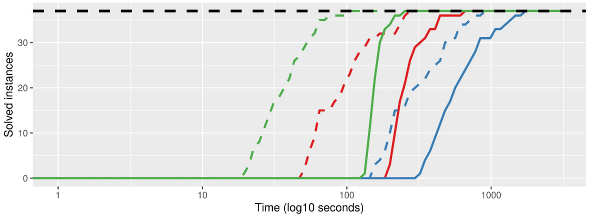
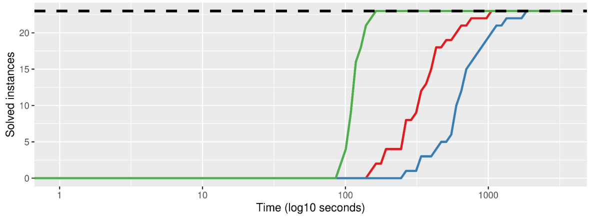
60 instances of the inference problem: transcriptomics, kinetics, and fluxomics noise from 0% to 50%



Choice of linear solvers impacts MerrinASP performance

Impact of linear solvers' APIs

60 instances of the inference problem: transcriptomics, kinetics, and fluxomics noise from 0% to 50%

	Satisfiable instances	Unsatisfiable instances
Core-carbon		
Medium-scale		

- - Total number of instances

Gurobi API

— Optlang

— GurobiPy (official)

— PuLP

Enumeration

- - 1 model

— 100 models or unsatisfiable

Not all linear solver APIs are adapted to successive solving and constraint updates

MerrinASP - Conflict generation

	Status	Solving methods	Number of calls to linear solvers	Number of refinements
Core-carbon	Satisfiable	Quantifier elimination	937 +/- 111	5 +/- 1
		CEGAR	**501 +/- 41**	6 +/- 1
	Unsatisfiable	Quantifier elimination	669 +/- 221	9 +/- 4
		CEGAR	**252 +/- 54**	9 +/- 4
Medium-scale	Satisfiable	Quantifier elimination	17 957 +/- 5 032	41 +/- 16
		CEGAR	**3 548 +/- 2 184**	21 +/- 11
	Unsatisfiable	Quantifier elimination	7 480 +/- 4 673	17 +/- 8
		CEGAR	**1 155 +/- 307**	13 +/- 3

CEGAR-based method reduce linear solver calls up to a factor of 7