Hybrid satisfiability methods for the inference of boolean regulations controlling metabolic networks

Kerian Thuillier

COLLEGE DOCTORAL BRETAGNE
MATHS, TELECOMS
INFORMATIQUE, SIGNAL
SYSTEMES, ELECTRONIQUE

Université de Rennes

THÈSE DE DOCTORAT DE

L'UNIVERSITÉ DE RENNES

ÉCOLE DOCTORALE N° 601
Mathématiques, Télécommunications, Informatique, Signal, Systèmes, Électronique
Spécialité : Informatique

Par
Kerian THUILLIER

Méthodes de Satisfiabilité Hybrides pour l'Inférence de Régulations Booléennes Contrôlant des Réseaux Métaboliques

Thèse présentée et soutenue à Rennes, le 27 septembre 2024
Unité de recherche : IRISA UMR 6074 – Équipe Dyliss

Rapporteurs avant soutenance :

François FAGES	Directeur de recherche, Centre INRIA Saclay
Simon de GIVRY	Chargé de recherche - HDR, INRAE – Toulouse

Composition du Jury :

Présidente :	Emmanuelle BECKER	Professeure des universités, Univ. Rennes – IRISA
Examinateurs :	Emmanuelle BECKER	Professeure des universités, Univ. Rennes – IRISA
	François FAGES	Directeur de recherche, Centre INRIA Saclay
	Simon de GIVRY	Chargé de recherche - HDR, INRAE – Toulouse
	Misbah RAZZAQ	Chargée de recherche, INRAE – Tours
	Laurent TOURNIER	Chargé de recherche, INRAE – Jouy-en-Josas
Dir. de thèse :	Anne SIEGEL	Directrice de recherche, CNRS – IRISA
Co-dir. de thèse :	Loïc PAULEVÉ	Directeur de recherche, CNRS – LaBRI

Remerciements

Tout d'abord, je tiens à remercier François Fages et Simon de Givry pour le temps passé à lire et rapporter ce manuscrit. Je souhaite également remercier Emmanuelle Becker d'avoir accepté de présider mon jury, et Misbah Razzaq et Laurent Tournier d'avoir accepté d'être dans mon jury. Merci à tous pour nos échanges, vos remarques et l'intérêt que vous avez porté à mes travaux lors de cette soutenance et au long de ma thèse.

Je tiens ensuite à remercier mes encadrants Anne Siegel et Loïc Paulevé.

Anne, merci de m'avoir proposé ce stage de fin de licence à l'interface entre ASP et la programmation linéaire. À ce moment-là, je n'aurais jamais imaginé que 6 ans plus tard je soutiendrai une thèse sur ces méthodes. Merci pour toute l'aide que tu m'as apportée lors de ces trois dernières années pour apprendre à gérer mon stress au travail et avoir tout mis en œuvre pour que je puisse être dans les meilleures conditions possibles. Je te remercie également pour tout le temps que tu as passé en "mode cruche" pour extirper et rendre compréhensibles toutes les idées que j'avais en tête. Ces travaux de thèse seraient encore moins digestes si tu ne l'avais pas fait.

Loïc, merci de ton accueil lors de mes séjours à Bordeaux. C'est toujours compliqué pour moi de sortir de ma zone de confort, mais ça a toujours été un plaisir de venir travailler avec toi en *vrai* (c'était quand même plus sympa qu'en visio !). Merci pour tous les moments de convivialité et les parties de *Magic*.

De manière générale, je vous remercie tous les deux pour votre bienveillance, pour m'avoir permis de travailler dans les meilleures conditions possibles et pour m'avoir soutenu avec empathie pendant ces trois années. Merci de m'avoir fait confiance pour cette thèse, et de m'avoir poussé toujours plus loin dans mes retranchements et vers le perfectionnisme. Vous avez fini par réussir à me faire me rendre compte de la quantité et de la qualité des travaux que j'ai réalisés pendant ces trois ans (ce qui n'était pas évident, le syndrome de l'imposteur n'est jamais très loin). Si cette thèse était à refaire avec vous, je n'hésiterais pas le moins du monde !

Je souhaiterais également remercier tout le groupe de travail de MERRIN: Caroline Baroukh, Alexander Bockmayr et Ludovic Cottret. Je vous remercie de m'avoir suivi durant toute ma thèse et d'avoir pris le temps de m'expliquer toutes les notions de biologie dont j'avais besoin pour travailler. Mon seul regret est que l'on n'ait pas réussi à tous se voir en personne. J'espère pouvoir continuer à travailler avec vous encore longtemps !

Je remercie également tous les membres des équipes de Symbiose (Dyliss, Genscale, GenOuest). Merci pour votre bonne humeur au quotidien, les pauses café, les séminaires au vert, tous les échanges liés au travail ou non. J'ai fait tous mes stages et ma thèse parmi vous, ça fait 6 ans que je vous côtoie, cela va vraiment me faire bizarre de vous quitter. Vous avez mis la barre très haute pour les prochaines équipes !

Merci Sandra Romain et Francesca Brunetti de m'avoir supporté comme co-bureau. Merci pour votre bonne humeur, tous nos échanges sérieux et moins sérieux (on était quand même doués pour procrastiner tous les trois, de vrais "maledetti monelli"[1]), et les pauses café.

Merci à toute l'équipe de *Science en Court[t]s*: Roland Faure, Khodor Hannoush, Sandra Romain, Baptiste Ruiz. Je suis très heureux que l'on ait pu réaliser ce court-métrage[2] tous les cinq, et d'avoir appris à vous connaître au cours de ces trois années.

Merci à Victor Épain, Matthieu Bouguéon et Lucas Robidou pour votre amitié et d'avoir été là dans les moments compliqués en m'aidant à me changer les idées (parties de *Magic*, sorties cinéma, soirées jeux de société, visio-pauses, etc.). Sans vous, je ne sais pas comment j'aurais survécu au stress et à mon syndrome de l'imposteur.

Je tiens également à remercier Rumen Andonov pour m'avoir fait participer à mon premier congrès à Montpellier (ROADEF 2020) alors même que je n'étais qu'en première année de Master. Merci de m'avoir fait confiance pour les stages (Licence et Master 1) et les missions d'enseignement, pour tous nos échanges et pour ta bonne humeur constante.

Évidemment, je remercie également ma famille et mes ami·e·s pour tout le soutien qu'ils m'ont apporté durant ces neuf années d'études. Vous m'avez permis de réaliser mes études dans les meilleures conditions possibles, je ne serais pas arrivé jusque-là sans vous.

Je remercie également mes deux boules de poils préférées (oui, je parle de mes chats !) : Ragnarok et Rapsodie. Je ne pouvais pas rêver de meilleurs *coding ducks* et peluches anti-stress.

Pour finir, je tiens à te remercier, Margaux. Merci d'être avec moi depuis toutes ces années. Je ne serai clairement pas allé aussi loin sans ton soutien. Je t'aime.

[1] Merci à Francesca de m'avoir appris ces quelques mots d'italien fort utiles !
[2] https://www.youtube.com/watch?v=9taXJ3P91YM

Résumé en Français

Ce manuscrit de thèse est rédigé sous la forme d'une « thèse par articles », et est rédigé en anglais. Il commence par un résumé en français du contenu du manuscrit, qui est suivi d'une introduction et une présentation de l'état de l'art du domaine (Chapitre I). Le manuscrit est rédigé en anglais à partir de l'introduction. Le chapitre II propose une définition générale des problèmes résolus dans le manuscrit. Les chapitres III à V sont basés sur des publications. Ils suivent la même structure : une introduction motivant le problème traité dans le chapitre, une description des contributions principales de la publication, une discussion étendant les résultats de la publication, et enfin la publication. Le manuscrit se termine avec une conclusion et une présentation des perspectives.

1 Modélisation et Inférence des Systèmes Biologiques

Biologie des systèmes. La biologie computationnelle, ou *bio-informatique*, est une discipline à l'interface entre les sciences du vivant et l'informatique (Kitano, 2002a). Durant les dernières décennies, de nouvelles techniques d'observation et de mesure haut-débit des cellules ont induit un changement majeur de paradigme en biologie et bio-informatique. La quantité de données collectées a permis l'émergence de la biologie des systèmes, un domaine de la bio-informatique s'intéressant aux mécanismes biologiques d'un point de vue systémique (Kitano, 2002b). Un des enjeux majeurs de la biologie des systèmes est l'intégration des données expérimentales, en particulier « omiques », pour générer de nouvelles connaissances, notamment expliquer le comportement et prédire la réponse des systèmes biologiques. Les données « omiques » sont un ensemble de données biologiques caractérisant et quantifiant des molécules d'intérêt dans les cellules (*e.g.* génomique, métabolomique).

Systèmes biologiques complexes. Les systèmes biologiques sont traditionnellement considérés comme des systèmes complexes composés de nombreux mécanismes biologiques interconnectés les uns avec les autres. Ces mécanismes sont regroupés en différentes échelles de processus, allant de la transcription des gènes d'une séquence d'ADN aux interactions entre populations de bactéries. Chaque échelle est souvent représentée indépendamment des autres avec sa propre dynamique et échelle de temps (Walpole et al., 2013).

En particulier, le système de régulation et le métabolisme sont deux échelles d'intérêt en biologie des systèmes. Le métabolisme transforme des nutriments en composés, appelés *métabolites*, nécessaires à la production de biomasse et

d'énergie. Ces transformations d'ensembles de métabolites vers d'autres ensembles de métabolites ont lieu sous l'activité de protéines, appelées *enzymes*, catalysant des réactions bio-chimiques. La production des enzymes est elle-même contrôlée par une cascade de régulation impliquant d'autres protéines, des métabolites et des facteurs abiotiques. Nous savons depuis le début des années 1940, et les travaux de Jacques Monod (Monod, 1942), que ces deux systèmes sont fortement interconnectés et qu'ils doivent tout deux être pris en compte pour expliquer certaines dynamiques des bactéries observées en laboratoire. En effet, certains métabolites produits par le métabolisme peuvent également bloquer ou induire des signaux de régulations, qui eux-mêmes peuvent bloquer ou induire l'expression de gènes, impactant l'activité du métabolisme.

Une modélisation indépendante. Malgré ces intéractions entre les échelles métabolique et de régulation, la plupart des méthodes de simulation du métabolisme et du système de régulation ne considèrent que l'une des deux échelles.

À l'*échelle métabolique*, la dynamique est classiquement abstraite par des systèmes algébriques différentiels selon une sémantique de flux, nommée *Flux Balance Analysis* (FBA) : les transformations de métabolites sont abstraites par le taux d'activité des réactions selon des hypothèses d'état stable et d'optimisation de la production de biomasse (Orth et al., 2010). En effet, il est couramment supposé que les cellules ont évolué pour optimiser certaines fonctions biologiques, *e.g.* leur croissance (Feist and Palsson, 2010). La FBA abstrait la dynamique du métabolisme comme un problème d'optimisation linéaire.

À l'inverse, à l'*échelle de la régulation*, la dynamique du système est discrétisée : un gène est soit actif, soit inactif, et son état est dépendant de la présence ou absence de protéines et/ou métabolites (Kauffman, 1969; Thomas, 1973; Wang et al., 2012). En pratique, la dynamique du réseau de régulation est modélisée par un réseau booléen, associant à chaque gène une fonction booléenne définissant son état. Ces fonctions booléennes, ou *règles de régulation booléennes*, sont fonction de l'état (actif ou inactif) de sous-ensembles de composés (*e.g.* gènes, protéines, métabolites) pouvant affecter l'expression des gènes. Selon ce formalisme, la mise à jour d'un état de régulation (un vecteur booléen) se calcule en appliquant toutes (synchrone), ou une partie (asynchrone), les règles de régulation du réseau booléen.

Il existe de nombreux formalismes et outils de simulation permettant de coupler la dynamique basée-flux du métabolisme avec la dynamique discrète du système de régulation (Moulin et al., 2021). Cependant, l'utilisation de ces formalismes de simulation est limitée par la disponibilité des modèles métaboliques régulés. En effet, il existe très peu de modèles métaboliques régulés disponibles dans la littérature. La plupart des modèles existants ne suivent pas les formalismes standards et sont inutilisables avec les outils de simulation actuels.

Modèles métaboliques régulés. Bien qu'il existe des méthodes de reconstruction de modèle métabolique (Thiele and Palsson, 2010) et d'inférence de règles de régulation (Videla et al., 2017; Chevalier et al., 2020; Ostrowski et al., 2016; Vaginay et al., 2021; Žiga Pušnik et al., 2022) à partir de données omiques, aucune méthode ne permet d'inférer les règles de contrôle du système de régulation sur le métabolisme et les règles de rétroaction de l'activité métabolique sur le système de régulation. Actuellement, les règles de contrôle et de rétroaction doivent être reconstruites manuellement. La reconstruction manuelle de ces règles a notamment été faite pour des modèles des bactéries comme *Escherichia coli* (Covert and Palsson, 2002; Covert et al., 2004), *Ralstonia solanacearum* (Peyraud et al., 2018) et *Bacillus subtilis*[3] (Tournier et al., 2017). L'absence de méthode d'inférence automatique de ces règles de régulation limite fortement le développement de nouveaux modèles de métabolismes régulés.

Objectif de la thèse. Les travaux réalisés aux cours de cette thèse visent à résoudre ce problème. Ce manuscrit présente de nouveaux formalismes, et outils, d'inférence de règles de régulation booléennes contrôlant le métabolisme à partir de données *omiques* et de connaissances biologiques *a priori*.

2 L'Inférence de Règles de Régulation dans la Littérature

Cette section résume l'état de l'art sur les formalismes de modélisation et d'inférence de modèles métaboliques régulés présenté Chapitre I.

2.1 Formalisme de Simulation des Modèles de Métabolisme Régulé

Modélisation de la dynamique couplée. Un modèle métabolique régulé est composé de deux éléments : un *réseau métabolique*, pour l'échelle du métabolisme ; et un *réseau booléen*, pour l'échelle de régulation. Les règles de régulation du réseau booléen peuvent être fonction de l'activité des réactions du métabolisme et la disponibilité des métabolites environnementaux (**règle de rétroaction**). De même, certaines réactions du métabolisme ont une règle de régulation dans le réseau booléen (**règle de contrôle**). Une réaction inhibée par le système de régulation a une activité nulle dans le métabolisme (*i.e.* aucune enzyme ne catalyse la réaction).

Le formalisme rFBA permet de modéliser la dynamique hybride des modèles métaboliques régulés (Covert et al., 2001). Le principe de la rFBA est de diviser la simulation du système en pas-de-temps, et d'alterner successivement : **(1)** une mise

[3] La rétroaction du métabolisme n'est pas prise en compte dans le modèle métabolique régulé de *Bacillus subtilis*.

à jour synchrone de l'état de régulation, et **(2)** résoudre les équations linéaires de la FBA après avoir fixé toutes les réactions inhibées à zéro.

2.2 Construction de Modèle de Métabolisme Régulé

Pour construire un modèle métabolique régulé, il est nécessaire de construire le réseau métabolique, et le réseau booléen, avec les règles de contrôle et de rétroaction. Actuellement, aucune méthode de la littérature ne permet d'inférer automatiquement les règles de régulation de contrôle et de rétroaction. Cependant, il existe des protocoles et méthodes pour la reconstruction/inférence des réseaux métaboliques et booléens.

Construction de réseaux métaboliques. Les réseaux métaboliques sont construits à partir de données génomiques, métabolomiques et fluxomiques (Thiele and Palsson, 2010). En particulier, les équations FBA, modélisant la dynamique du métabolisme, sont calibrées à partir de données cinétiques et fluxomiques. Dans ce manuscrit, nous considérons les réseaux métaboliques des modèles étudiés comme des entrées de nos méthodes.

Inférence de réseaux booléens. Pour les réseaux booléens, de nombreuses méthodes ont été développées pour inférer des réseaux booléens à partir de série temporelle de données d'expression (transcriptomiques ou protéomiques) et de connaissances *a priori*. Ces connaissances prennent la forme d'ensemble d'interactions autorisées entre les gènes, protéines et métabolites. La règle de régulation f_n d'un composé n est supportée par un ensembles d'interactions $\{m_1 \to n, m_2 \to n\}$ si f_n est uniquement fonction de m_1 et m_2. Notons que ces interactions peuvent être signées. Elles sont positives $m \to^+ n$ si m permet l'activation de n, et négatives $m \to^- n$ si m inhibe n.

Les méthodes d'inférence de réseaux booléens formulent le problème d'inférence comme un problème d'optimisation combinatoire (Videla et al., 2017; Ostrowski et al., 2016; Chevalier et al., 2020; Vaginay et al., 2021) ou de programmation mixte en nombre entier (Terfve et al., 2012). En particulier, les méthodes basées sur l'optimisation combinatoire permettent d'énumérer l'ensemble des réseaux booléens compatibles avec les données d'entrée, un critère important pour les biologistes. Ces derniers ont besoin d'avoir une vue d'ensemble de l'espace des possibles afin de planifier leurs protocoles expérimentaux.
Notons également qu'il existe des méthodes d'inférence stochastique (Trinh and Kwon, 2021; Gao et al., 2020; Liu et al., 2021; Barman and Kwon, 2020), mais ces méthodes ne prennent pas en entrée des interactions et ne peuvent pas énumérer l'ensemble des solutions. Nous ne les considérons pas dans cette thèse.

Limites. L'inconvénient de ces méthodes est qu'elles ne considèrent pas la dynamique et l'impact du métabolisme sur le système de régulation. Elles infèrent des réseaux booléens en ne considérant que la dynamique discrète du réseau de régulation. Elles ne permettent donc pas de capter les règles de contrôle et de rétroaction.

3 Contributions : Inférence de Réseaux Booléens Contrôlant des Réseaux Métaboliques

Cette section résume les chapitres II à V regroupant la formalisation du problème d'inférence et les différentes méthodes de résolution développées pour le résoudre.

Définition générale. Pour inférer les règles de contrôle et de rétroaction, il est nécessaire de prendre en compte la dynamique hybride des modèles métaboliques régulés. Dans le **Chapitre II**, nous proposons une **définition générale du problème d'inférence** de réseaux booléens contrôlant le métabolisme à partir de données omiques. Cette définition prend la forme d'un problème d'optimisation sous contraintes quantifiées.

Ici, nous considérons trois types de données omiques : transcriptomiques, cinétiques et fluxomiques. Les données transcriptomiques sont des données mesurant l'expression des gènes. Ce type de données est communément employé pour inférer des réseaux booléens modélisant des réseaux de régulation. Les données cinétiques et fluxomiques sont quant à elles des mesures de l'activité du métabolisme, respectivement, les concentrations de métabolites environnementaux et une quantification de l'activité des réactions. Les données cinétiques et fluxomiques sont employées pour reconstruire et calibrer les équations FBA modélisant le métabolisme.

En considérant le formalisme rFBA pour modéliser la dynamique du métabolisme régulé, nous formulons le problème d'inférence comme :

Entrées :

1: un réseau métabolique ;
2: des connaissances biologique *a priori* sous la forme d'un ensemble d'interactions possibles entre les gènes, protéines, métabolites et réactions ;
3: des séries temporelles de données cinétiques, fluxomiques ou transcriptomiques.

Sorties : l'ensemble des réseaux booléens tels que :

1: le réseau booléen contrôlant le réseau métabolique d'entrée admet des simulations rFBA *minimales* qui sont compatibles avec chaque série temporelle ;

2: le réseau booléen est supporté par les connaissances biologiques.

Formellement, cette définition du problème d'inférence prend la forme d'un **problème d'optimisation hybride combinant des contraintes logiques et des contraintes linéaires quantifiées**. En pratique, nous avons dérivé trois formulations du problème d'inférence : une relaxation booléenne (Chapitre III), et deux formulations hybrides (Chapitres IV et V).

3.1 Relaxation Booléenne du Problème d'Inférence

Cette section résume le chapitre III dont le contenu est basé sur la papier Thuillier et al. (2021).

Relaxation booléenne. Dans le **Chapitre III**, nous introduisons une **relaxation booléenne du problème d'inférence** (Thuillier et al., 2021). Cette relaxation est basée sur une sur-approximation booléenne des équations linéaires de la FBA permettant de définir une abstraction booléenne du formalisme rFBA. En pratique, la formulation relaxée du problème d'inférence est un **problème de satisfiabilité logique avec deux niveaux de quantificateurs (2-QBF)**.

Programmation par ensembles réponses. Pour résoudre ce problème 2-QBF, nous avons utilisé la *programmation par ensembles réponses* (ASP) (Baral, 2003) et la méthode de *saturation* (Eiter et al., 2009; Gebser et al., 2011). La méthode de saturation permet de résoudre efficacement les problèmes 2-QBF en exploitant la sémantique stable d'ASP et la sémantique des contraintes logiques disjonctives. L'encodage ASP du problème d'inférence relaxé est décrit en Annexe B.2.

Application. L'application de notre méthode à deux modèles dérivés d'un modèle du métabolisme central du carbone d'*Escherichia coli*[4] (Covert et al., 2001) a donné des résultats prometteurs, mais a également mis en évidence certaines limites. Bien que la méthode ait permis d'inférer des réseaux booléens reproduisant exactement les simulations rFBA utilisées pour générer les séries temporelles d'entrée, elle a également conduit à l'inférence de réseaux *faux-positifs*. Il y a environ 50% de réseaux faux-positifs d'inférés pour l'un des cas d'étude. Ces faux-positifs sont dus à notre sur-approximation booléenne des équations de la FBA. Cette dernière génère des états métaboliques booléens stationnaires qui n'ont pas de contrepartie dans les équations FBA. Malgré cela, cette abstraction booléenne est à la base de toutes les méthodes de résolution décrites dans ce manuscrit.

[4]Scripts disponibles sur *GitHub* : https://github.com/bioasp/boolean-caspo-flux.

3.2 Formulation Hybride du Problème d'Inférence

Cette section résume le chapitre IV dont le contenu est associé au papier Thuillier et al. (2022).

Formulation hybride. Pour ne pas inférer de réseaux booléens *faux-positifs*, il est nécessaire d'intégrer la vérification des équations linéaires de la FBA directement dans le processus d'inférence. La difficulté réside dans la nécessité de combiner la dynamique discrète des réseaux booléens et la dynamique linéaire du métabolisme. Dans le **Chapitre IV**, nous présentons **un schéma de résolution hybride**, couplant la programmation logique et la programmation linéaire, pour résoudre la formulation hybride du problème d'inférence sans passer par des sur-approximations booléennes (Thuillier et al., 2022).

Schéma de résolution hybride. Ce schéma de résolution repose sur des méthodes de propagation de contraintes et de généralisation de contre-exemples, communément utilisées dans les solveurs *satisfiabilité modulo théorie* (SMT) (Barrett and Tinelli, 2018). En pratique, il permet d'intégrer la vérification des équations de la FBA et le critère de maximisation de la croissance à ASP (Ostrowski and Schaub, 2012; Banbara et al., 2017).

Ce schéma de résolution peut être résumé en trois grandes étapes. Tout d'abord, on cherche une solution au problème d'inférence relaxé, et on extrait tous les états métaboliques booléens stationnaires qui sont générés. Ensuite, on vérifie pour chaque état s'il existe une solution de la FBA équivalente, et que la croissance optimale prédite par la FBA colle avec les observations. Si oui, le réseau booléen inféré est solution. Sinon, le réseau booléen est un *contre-exemple*. Le réseau est rejeté et de nouvelles contraintes ASP sont générées. Ce processus itératif continue jusqu'à ce que le problème soit prouvé non-satisfiable ou que toutes les solutions aient été énumérées.

Généralisation de contre-exemple. Les contraintes ASP générées pour chaque contre-exemple sont très importantes pour la résolution du problème et permettent le passage à l'échelle de notre méthode. Ces contraintes permettent de généraliser les contre-exemples, et ainsi éviter de générer des réseaux booléens candidats qui sont sûr d'échouer la vérification linéaire. Ces contraintes sont générées en exploitant une propriété monotone liant les ensembles de réactions inhibées et la croissance optimale selon la FBA. Intuitivement, la propriété énonce que "inhiber une réaction ne peut pas permettre d'augmenter la croissance".

Implémentation et validation. Ce schéma de résolution a été implémenté dans un outil dédié à l'inférence de réseaux booléens contrôlant des réseaux métaboliques :

MERRIN (https://github.com/bioasp/merrin).

Pour valider *MERRIN*, nous avons développé un protocole de génération de données omiques synthétiques, mais réaliste, à partir de simulations rFBA. Ce protocole permet de générer des séries temporelles bruitées (ou non) de données cinétiques, fluxomiques et transcriptomiques. Les séries temporelles générées peuvent être composées de tout sous-ensemble de ces trois types de données omiques. À l'aide de ce protocole, nous avons généré un benchmark de 240 instances avec différentes combinaisons de type de données et différents niveaux de bruit (0% à 50%). Les instances sont générées à partir de cinq simulations rFBA d'un modèle de métabolisme central du carbone d'*Escherichia coli* (Covert et al., 2001).

Sur ce benchmark, *MERRIN* permet d'inférer des réseaux booléens plus petit que celui de référence, *i.e.* pour ces réseaux, toutes les règles de régulation ne sont pas inférées. Malgré cela, ces réseaux *minimaux* permettent de reproduire exactement les simulations rFBA utilisées pour générer le benchmark. *MERRIN* permet donc d'inférer des modèles plus parcimonieux que ceux de la littérature, tout en expliquant les mêmes comportements. De manière générale, nous avons constaté que les réseaux inférés par *MERRIN* retrouvent exactement les simulations rFBA d'entrée, ou à un point de temps prêt, dès lors que l'on utilise des données cinétiques et transcriptomiques avec moins de 20% de bruit. Cela montre donc qu'il est possible d'inférer des réseaux booléens de régulation, et notamment les règles de contrôle et de rétroaction, à partir de données cinétiques et transcriptomiques.

3.3 Optimisation Combinatoire sous Contraintes Linéaires Quantifiées

Cette section résume le chapitre V dont le contenu est associé au papier Thuillier et al. (2024).

Problèmes OPT+qLP. La formulation hybride du problème d'inférence, résolu par *MERRIN*, est un exemple de problèmes d'optimisation combinatoire sous contraintes linéaires quantifiées (OPT+qLP). Dans ces travaux, on se restreint à un seul niveau de quantificateurs linéaires. Dans le **Chapitre V**, nous introduisons **une méthode générique de résolution des problèmes OPT+qLP** basée sur une méthode de *raffinement d'abstraction guidé par les contre-exemples* (*Counter-Example Guided Abstract Refinement* – CEGAR) (Clarke et al., 2003).

Méthode CEGAR. La méthode CEGAR est une méthode générique permettant de combiner facilement des solveurs, notamment des solveurs logiques et des solveurs linéaires. Cette méthode a déjà été employée pour résoudre des problèmes d'optimisation hybride (Janota et al., 2016; Brummayer and Biere, 2008; Barrett and Tinelli, 2018), mais pas pour résoudre des problèmes OPT+qLP.

La méthode CEGAR est assez similaire au principe de résolution et génération de contraintes utilisé dans *MERRIN*. Une sur-approximation booléenne du problème OPT+qLP est résolue avec un solveur SAT ou ASP. Si cette abstraction est non-satisfiable, alors le problème OPT+qLP l'est aussi. Sinon, un modèle de l'abstraction booléenne est trouvé. Ce modèle est une solution au problème OPT+qLP s'il satisfait les contraintes linéaires quantifiées. Sinon, c'est un *contre-exemple*, et l'abstraction est raffinée en ajoutant de nouvelles contraintes dérivées du contre-exemple. Ce processus itératif continue jusqu'à ce que le problème OPT+qLP soit démontré non-satisfiable ou que toutes les solutions aient été énumérées.

Généralisation de contre-exemple. La généralisation des contre-exemples, et donc des nouvelles contraintes, repose sur une propriété monotone liant la structure des problèmes d'optimisation linéaire à leur optimum. Intuitivement, cette propriété énonce que "ajouter une nouvelle contrainte linéaire à un problème d'optimisation linéaire ne peut pas augmenter (*resp.* diminuer) son maximum (*resp.* minimum)". La généralisation des contre-exemples raisonne donc sur les ensembles de contraintes des problèmes d'optimisation linéaire. En pratique, pour chaque contre-exemple, nous allons calculer les *meilleurs* ensembles de contraintes linéaires avant de générer les contraintes, *i.e.* on cherche à maximiser (ou minimiser suivant la situation) la taille des contre-exemples. Cela permet aux contraintes générées de filtrer plus efficacement l'espace des solutions tout en conservant un coût de calcul raisonnable.

Implémentation et benchmark. Nous avons implémenté ce schéma de résolution dans le solveur générique *MerrinASP* (`https://github.com/kthuillier/merrinasp`). Il étend le solveur ASP *clingo* avec des contraintes linéaires ayant un niveau de quantificateur.

Nous avons comparé les performances de *MerrinASP* avec *clingo-lpx*, un solveur qui étend ASP avec des contraintes linéaires sans quantificateur. Comme aucun benchmark de problèmes OPT+qLP existe, nous avons utilisé un benchmark issu du problème d'inférence sur deux modèles métaboliques régulés, le modèle de métabolisme central du carbone (Covert et al., 2001) et un modèle moyenne échelle (Covert and Palsson, 2002)[5]. Un protocole d'élimination de quantificateurs a été utilisé pour reformuler les instances fournies à *clingo-lpx*. Ce protocole repose sur une formulation des problèmes d'optimisation linéaire à l'aide du théorème de dualité forte. En pratique, nous avons constaté que *MerrinASP* est 10 fois plus rapide à résoudre les problèmes OPT+qLP que *clingo-lpx* avec élimination de quantificateurs.

De plus, résoudre le problème d'inférence hybride en l'encodant avec *MerrinASP*

[5]Notons que *MERRIN* ne passait pas à l'échelle, au niveau des temps de calcul, sur ce modèle grande échelle.

est plus performant que *MERRIN*. *MERRIN* n'a pas pu inférer de réseaux booléens sur les instances issues du modèle grande échelle en 24h, tandis qu'il faut environ 2min avec l'encodage *MerrinASP* pour en trouver un premier (plus de 150 000 réseaux sont inférés en 24h). Ce gain de performance repose principalement sur l'optimisation des contre-exemples avant la généralisation des contraintes, qui n'est pas implémenté dans *MERRIN*.

4 Conclusion

Cette thèse introduit différentes méthodes d'inférence de réseaux booléens contrôlant des réseaux métaboliques. Elle montre que le problème d'inférence peut être formulé comme un problème d'optimisation hybride combinant contraintes logiques et linéaires quantifiées. Considérer la dynamique linéaire du métabolisme pour inférer les réseaux booléens permet d'inférer automatiquement des réseaux booléens avec les règles de contrôle et de rétroaction. En particulier, nous montrons qu'il est possible d'inférer ces règles à partir de données cinétiques et transcriptomiques.

Cette thèse a également été l'occasion de constater que les méthodes de la littérature pour résoudre des problèmes d'optimisation hybride ne sont pas adaptées aux problèmes de la biologie des systèmes. En particulier, en biologie de systèmes, les problèmes sont des problèmes d'optimisation hautement combinatoires pour lesquels il est nécessaire d'énumérer toutes les solutions ou d'échantillonner l'espace des solutions. Pour palier à cela, nous avons dû développer de nouvelles méthodes de résolutions dédiées et adaptées à ces caractéristiques.

4.1 Publications

Les travaux présentés dans cette thèse ont tous été présentés en conférence et publiés :

— La relaxation du problème d'inférence et la méthode de résolution associée (Chapitre III) ont été présentées lors de la conférence internationale *Computational Methods in Systems Biology* (CMSB) en 2021, et publiées dans les actes de la conférence (Thuillier et al., 2021).

— La formulation basée-flux du problème d'inférence et la méthode de résolution hybride *MERRIN* (Chapitre IV) ont été présentées lors de la conférence européenne *European Conference on Computational Biology* (ECCB) en 2022, et publiées dans la revue *Bioinformatics* (Thuillier et al., 2022).

— La méthode de résolution des problèmes OPT+qLP a été présentée lors de la conférence nord américaine de l'*Association for the Advancement of Artificial Intelligence* (AAAI) en 2024, et publiée dans les actes de la conférence (Thuillier et al., 2024).

4.2 Ressources

Outils et benchmarks. Durant cette thèse, nous nous sommes assurés que tous nos résultats étaient reproductibles. Ainsi, les contributions de la thèse sont toutes associées à des outils et des dépôts en ligne :

— L'encodage ASP du problème d'inférence relaxé, les scripts et les données utilisés dans le chapitre III sont disponibles au lien : https://github.com/bioasp/boolean-caspo-flux.

— L'outil *MERRIN* dédié à l'inférence de réseaux booléens contrôlant le métabolisme (formulation hybride) (Chapitre IV) est disponible au lien https://github.com/bioasp/merrin. Les scripts et données pour reproduire le benchmark et les résultats sont également disponibles : https://github.com/bioasp/merrin-covert.

— Le solveur générique *MerrinASP* (Chapitre V) permettant de résoudre les problèmes d'optimisation combinatoire sous contraintes linéaires quantifiées est disponible au lien https://github.com/kthuillier/merrinasp. Les scripts et données du benchmark sont également disponibles : https://zenodo.org/records/10361533.

Modèles métaboliques régulés. En plus des méthodes et outils présentés, cette thèse a été l'occasion de réactualiser trois modèles métaboliques régulés de *Escherichia coli* disponible dans la littérature (Covert et al., 2001; Covert and Palsson, 2002; Covert et al., 2004). Ces modèles sont décrits en annexes (Annexe A), les fichiers sont disponibles à https://github.com/kthuillier/regulated-metabolic-models.

Contents

Introduction

This manuscript is written in a "thesis on publications" format. It starts with a quick summary of the manuscript content written in French. Afterward, the manuscript is written in English. This summary is followed by an introduction and a review of the state of the art (Chapter I). Then, it presents a formal definition of the inference of Boolean regulation controlling metabolic networks problem (Chapter II). Chapters III to V are based on individual publications and address different formulations of the inference problem. They follow the same structure: an introduction motivating the chapter's content, ***(ii)*** *a detailed description of the publication's main contributions,* ***(iii)*** *a discussion extending the publication's results with new benchmarks,* ***(iv)*** *the publication. The manuscript ends with a conclusion discussing future perspectives.*

Systems biology. Computational biology, or *bioinformatics*, is a discipline at the interface of life sciences and computer science (Kitano, 2002a). During the last decades, advances in high-throughput observation and measurement techniques have led to a paradigm shift in biology and bioinformatics. The amount of data collected has led to the emergence of *systems biology*, a field of bioinformatics focused on understanding biological mechanisms from a systems perspective. Systems biology considers biological processes as a whole rather than isolated parts. A major challenge in systems biology is to integrate experimental data, especially the so-called *omics* data, to generate new insights that explain cell behaviors and predict their responses to environmental changes (Joyce and Palsson, 2006).

Complex biological systems. Traditionally, biological systems are considered complex systems composed of many interconnected mechanisms that operate on different timescales. These mechanisms range from gene transcription within DNA sequences to interactions between populations of bacteria. Typically, each scale is represented independently, with distinct dynamics formalism and timescales specific to that scale (Walpole et al., 2013).

In particular, the regulatory system and metabolism are two scales of interest in systems biology. Since the early 1940s, following the work of Jacques Monod (Monod, 1942), it is known that these two systems are strongly interconnected and that both scales should be considered to explain and predict bacterial growth behaviors. Metabolism transforms nutrients into compounds, known as *metabolites*, which are essential for the cell to produce biomass and energy. These transformations occur through biochemical reactions catalyzed by special proteins, called *enzymes*, whose production is itself controlled by a cascade of regulations involving proteins, metabolites, and abiotic factors (*e.g.* temperature, pH). In

addition, byproduct metabolites of the metabolism can inhibit or induce regulatory signals, which can also inhibit or induce gene expression, and indirectly impact the metabolic activity.

Metabolic and regulatory scales are modeled separately. Despite the feedback and control interactions between the metabolic and regulatory scales, most simulation and inferring formalisms consider the dynamics of these scales separately.

At the *metabolic scale*, dynamics are typically abstracted using differential algebraic systems based on a flux semantic called flux balance analysis (FBA): metabolite transformations are represented by reaction rates under steady-state assumptions and biomass production optimization (Orth et al., 2010). The FBA models the metabolism dynamics as a linear optimization problem, whose variables are flux over reactions.

At the *regulatory scale*, system dynamics is discretized: a gene is either active or inactive, and its state is dependent on the presence or absence of proteins and/or metabolites (Kauffman, 1969; Wang et al., 2012). The cost of transcribing a protein from gene expression is considered negligible, so it is generally deemed sufficient to know that the protein is available without considering its concentration. Typically, the dynamic of the regulatory system is represented by Boolean networks.

Over the years, many formalisms and simulation tools have been introduced to integrate the flux-based dynamics of metabolism with the discrete dynamics of the regulatory system. However, the uses of these formalisms are limited by the availability of high-quality regulated metabolic models.

Regulated metabolic models. There exist methods for reconstructing metabolic models from genomics, metabolomics, and fluxomics data (Thiele and Palsson, 2010); and for inferring regulatory rules from expression data (Videla et al., 2017; Chevalier et al., 2020; Ostrowski et al., 2016; Vaginay et al., 2021; Žiga Pušnik et al., 2022). The bottleneck is to infer the feedback and control interactions between the metabolic and regulatory scales. Currently, these interactions must be manually reconstructed and curated, as it was done for the bacterium *Escherichia coli* (Covert and Palsson, 2002; Covert et al., 2004) or *Bacillus subtilis* (Tournier et al., 2017). The absence of automated inference methods significantly hinders the development of new regulated metabolic models, and thus, the use of hybrid simulation formalisms to accurately predict cell behaviors.

In this manuscript. The work presented in this thesis aims to address this issue by presenting new formalisms and solving methods to address the inference of Boolean regulatory rules controlling the metabolism from *omics* data and prior biological knowledge.

In **chapter I**, we review the **state-of-the-art formalisms** to model the activity of the metabolic and the regulatory layers, and to infer them '*in solo*'. We also review the **methods to address combinatorial optimization problems**, and their hybrid extensions with the *Answer Set Programming* (ASP).

In **chapter II**, we propose a **general definition for the inference problem** of Boolean regulatory rules that control metabolic networks from *omics* data and prior biological knowledge. This definition takes the form of a combinatorial optimization problem under quantified constraints. From this general definition, we derive three formulations of the inference problem that are solved in the next three chapters: a relaxed formulation (Chapter III) and two hybrid formulations (Chapters IV and V).

The **chapter III** is based on the paper Thuillier et al. (2021). Following state-of-the-art Boolean networks inference methods, we relax the general definition of the inference problem as **a Boolean satisfiability problem**, based on a Boolean abstraction of the metabolism dynamics. We present an **ASP-based implementation** to solve it, which we apply to a simplified model of core-carbon metabolism of *Escherichia coli*.

The **chapter IV** is based on the paper Thuillier et al. (2022). In this chapter, we present **a hybrid inferring workflow, and its implementation *MERRIN***, to solve the inference problem. *MERRIN* integrates the flux-based dynamics of the metabolism with the discrete dynamics of the regulatory layer to ensure that inferred Boolean regulatory rules are compatible with the input *omics* data. We **validate it and test its robustness on a comprehensive benchmark** that we generate from a core-carbon metabolism model of *Escherichia coli*.

The **chapter V** is based on the paper Thuillier et al. (2024). The formulation of the inference problem addressed in Chapter IV belongs to the class of combinatorial optimization problems under quantified linear constraints (OPT+qLP). In this chapter, we present a novel **generic solving framework to address OPT+qLP problems, and its implementation *MerrinASP***. We benchmark *MerrinASP* against state-of-the-art hybrid solvers on a benchmark of inference problem instances.

The final chapter concludes the manuscript by summarizing the thesis contributions and highlighting future perspectives. Along with the theoretical and software contributions described in the manuscript, we cleaned and updated three regulated metabolic networks of the literature. The descriptions of these networks are available in Appendix A.

I State of the Art

In this chapter

1 Modeling Biological Systems

In this manuscript, we focus on two scales of biological processes: the metabolic scale and the regulatory scale. Despite being interconnected, the metabolic and regulatory scales are mostly modeled separately, without accounting for their interactions.

In the next sections, we review the state-of-the-art formalisms for modeling the activity of the metabolic scale (Section 1.1) and of the regulatory scale (Section 1.2). In Section 1.3, we describe the formalisms that account for the combined activity of both scales.

1.1 Modeling the Activity of the Metabolic Scale

The metabolic scale encompasses all the biochemical reactions involved in the production of energy and biomass within the cell. Reactions are *fast* biological processes occurring in the order of milliseconds to seconds. A *reaction* is a transformation of one set of chemical components, called *metabolites*, into another. For example, the reaction `A + 2 B` $\rightarrow$ `3 C` represents the transformation of 1 molecule of A and 2 molecules of B into 3 molecules of C. A reaction is said *reversible* if it can transform back its products into their initial states; otherwise the reaction is said *irreversible*. A *stoichiometric coefficient* is the number of molecules produced or consumed by the reaction; in the previous reaction, the stoichiometric coefficient of B is 2.

Cells are typically composed of thousands of reactions organized into pathways. For instance, the most accurate metabolic models of *Escherichia coli* accounts for 1 877 reactions over 2 712 metabolites (Monk et al., 2017). The KEGG map of its metabolic pathways (Kanehisa and Goto, 2000) is shown Fig. 1.

1.1.1 Metabolic Networks

The metabolism is usually represented by metabolic networks (Edwards and Palsson, 1999; Gu et al., 2019). A metabolic network abstracts the set of reactions using graph representations, usually either as bipartite graphs (Bourqui et al., 2007; Schaub and Thiele, 2009; Frioux et al., 2019) or hypergraphs (Cottret and Jourdan, 2010; Julien-Laferrière et al., 2016). Both representations are equivalent.

In this manuscript, we consider the bipartite graph definition introduced in Frioux et al. (2019). The metabolic network is a bipartite graph composed of two sets of nodes, representing metabolites and reactions, that are linked by a stoichiometric matrix, *i.e.* the graph's incidence matrix. The stoichiometric matrix details how, and in which proportions, metabolites are transformed (produced/consumed) by

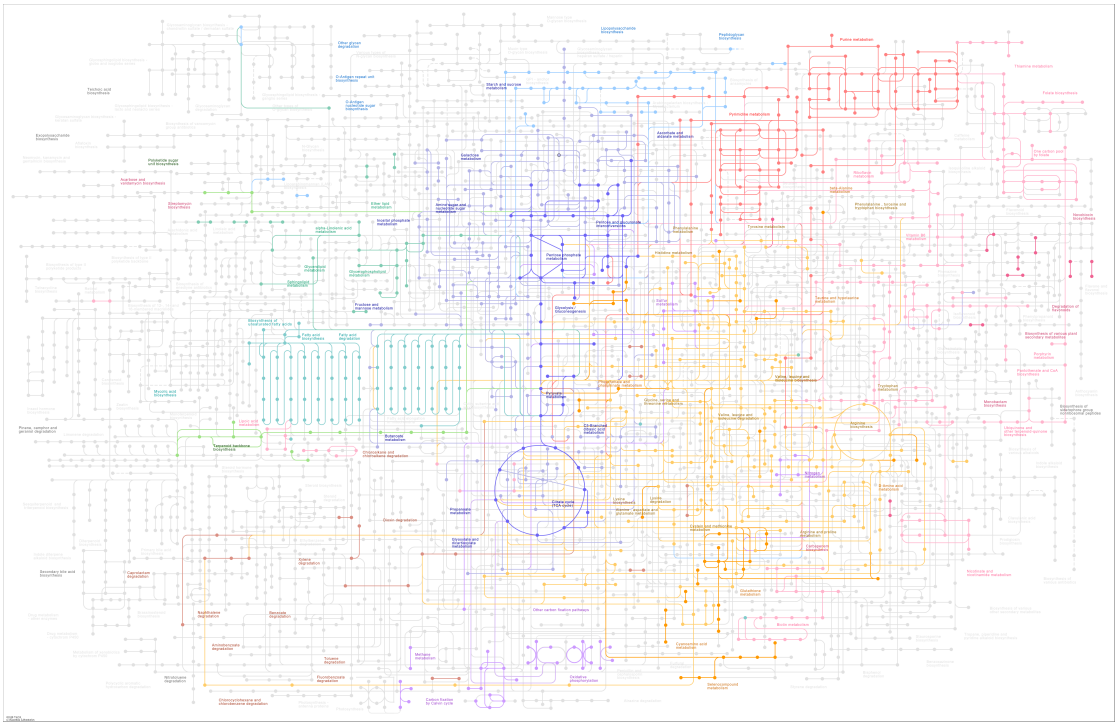


Figure 1 – KEGG map of metabolic pathways of the bacteria *Escherichia coli* K-12 MG1655 (June 26, 2024) (Kanehisa and Goto, 2000). Each color is associated with a specific biological pathway. Nodes are metabolites, and edges are associations between components, mostly related to reactions. Nodes and edges in light grey are components and interactions recorded in the KEGG database that do not belong to this bacteria.

each reaction. Note that, for the sake of clarity, the figures presented in this manuscript represent metabolic networks as weighted hypergraphs.

▶ **Definition 1.1: Metabolic network (Frioux et al., 2019)**

A *metabolic network* can be defined as a triple:

$$\mathcal{N} = (\mathcal{M} = \mathcal{M}_{\text{ext}} \cup \mathcal{M}_{\text{int}}, \mathcal{R}, S)$$

where $\mathcal{M}$ is a set of metabolites, and $\mathcal{R}$ is a set of reactions. The external metabolites $\mathcal{M}_{\text{ext}}$ are environmental metabolites, while the internal metabolite $\mathcal{M}_{\text{int}}$ are intracellular metabolites available in the cell cytosol.
S is the $|\mathcal{R}| \times |\mathcal{M}|$ stoichiometric matrix of real coefficients. It associates for each couple metabolite-reaction the stoichiometric coefficient of the metabolite for the reaction, *i.e.*, the relative quantities of the metabolite involved in the reaction.

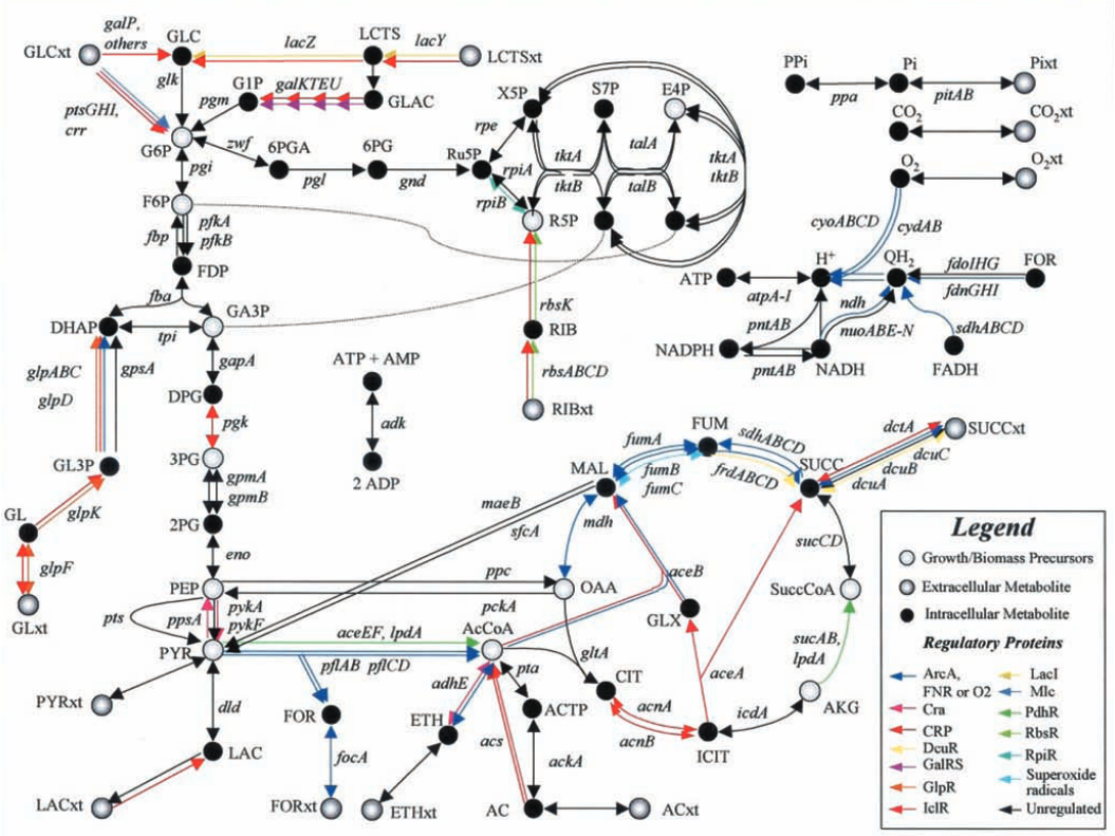


Figure 2 – Figure from Covert and Palsson (2002): Medium-scale metabolic network of the central metabolism of *Escherichia coli*. The network accounts for 113 reactions over 90 metabolites of which 77 are intracellular metabolites and 13 are environmental metabolites. Black circles are intracellular metabolites, grey circles are environmental metabolites, and white circles are intracellular metabolites taking part in the growth reaction. Arrows are reactions, and their labels are the genes encoding the enzymes catalyzing the reactions.

For the rest, we will call reactants (*resp.* products) of a reaction $r \in \mathcal{R}$ all the metabolites $m \in \mathcal{M}$ that are consumed (*resp.* produced) by the reaction, *i.e.* such that $S_{mr} < 0$ (*resp.* $S_{mr} > 0$). The reactions importing environmental metabolites in the cell are *exchange reactions*.

Example. A medium-scale metabolic network of the core metabolism of *Escherichia coli* introduced in (Covert and Palsson, 2002) is shown in Fig. 2. This network accounts for 113 reactions over 90 metabolites. Note that this network is quite small compared to the most precise metabolic networks of *Escherichia coli* currently available (1 877 reactions and 2 712 metabolites) (Monk et al., 2017).

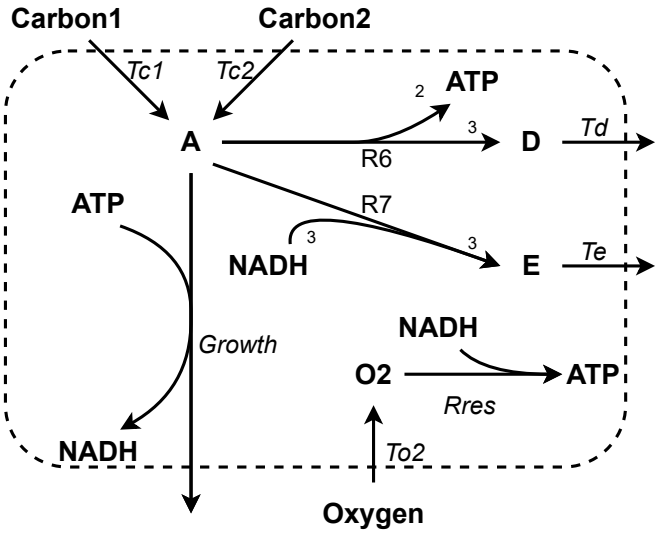


Figure 3 – Example of *toy* metabolic network represented as a hypergraph. Each node is a metabolite and each hyperedge is a reaction. All metabolites outside (*resp.* inside) the dotted round square are external (*resp.* internal) metabolites. Weights over reactions are stoichiometric coefficients. For example, the hyperedge R7 from $\{\mathrm{A}; \mathrm{NADH}\}$ to $\{\mathrm{E}\}$ models the reaction $\mathrm{A} + 3 \cdot \mathrm{NADH} \rightarrow \mathrm{E}$.

To simplify our examples, we introduce a *toy* metabolic network (Fig. 3). This network is a simplification of a model of core-carbon metabolism introduced in Covert et al. (2001). We use this *toy* metabolic network as a case study in Thuillier et al. (2021). This metabolic network is composed of 9 metabolites and 9 reactions. The internal metabolites are $\mathcal{M}_{\text{int}} = \{\text{A, D, E, O2, ATP, NADH}\}$, the environmental metabolites are $\mathcal{M}_{\text{ext}} = \{\text{Carbon1, Carbon2, Oxygen}\}$. The set of reactions is $\mathcal{R} = \{\text{Tc1, Tc2, To2, Td, Te, Growth, Rres, R6, R7}\}$. The three exchange reactions are Tc1, Tc2, and To2. The stoichiometric coefficients are also given in the figure. By default, they are set to 1, except for the reactions R6 and R7.

The stoichiometric matrix S of dimension $|\mathcal{R}| \times |\mathcal{M}|$ is such that $\forall\, r \in \mathcal{R}$, $\forall\, m \in \mathcal{M}$, $s_{mr} \in \mathbb{R}$ is the stoichiometric coefficient of the metabolite m for the reaction r.

The stoichiometric matrix for this example is defined as:

$$S = \begin{array}{c} \\ \textbf{Carbon1} \\ \textbf{Carbon2} \\ \textbf{Oxygen} \\ \textbf{A} \\ \textbf{D} \\ \textbf{E} \\ \textbf{O2} \\ \textbf{ATP} \\ \textbf{NADH} \end{array} \begin{array}{c} \begin{array}{ccccccccc} \textbf{Tc1} & \textbf{Tc2} & \textbf{To2} & \textbf{Td} & \textbf{Te} & \textbf{Growth} & \textbf{Rres} & \textbf{R6} & \textbf{R7} \end{array} \\ \left[\begin{array}{ccccccccc} -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & -1 & 0 & -1 & -1 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 3 \\ 0 & 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 1 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & -3 \end{array}\right] \end{array}$$

1.1.2 Flux Balance Analysis

There are many formalisms for modeling the activity of a metabolic network, including systems of ordinary differential equations and constraints-based modeling. A comprehensive review of these formalisms is available in Moulin et al. (2021).

The FBA is a constraint-based mathematical framework modeling the optimal distribution of fluxes in a metabolic network (Varma and Palsson, 1994; Orth et al., 2010). Each reaction $r \in \mathcal{R}$ is associated with a flux $v_r \in \mathbb{R}$ that models the reaction activity rate; usually measured in *millimoles per gram dry weight per hour* (mmol.gDW^{-1}.hr^{-1}). The flux v_r of a reaction r is bounded by a lower bound $l_r \in \mathbb{R}$ and an upper bound $u_r \in \mathbb{R}$. These two bounds ensure that the flux is biologically relevant. In practice, the bounds of exchange reactions are dependent on substrate concentrations (Varma and Palsson, 1994).

FBA assumptions. The FBA framework relies on two assumptions: **(i)** the steady-state assumption, and **(ii)** the growth optimality assumption. The steady-state assumption supposes that the concentrations of internal metabolites remain constant over time, *i.e.* $\forall\, m \in \mathcal{M}_{\text{int}}$, $\frac{d[m]}{dt} = 0$. In other words, for each internal metabolite, the production rate and consumption rate are balanced. Mathematically, it is expressed as the following linear constraint:

$$S \cdot v = 0$$

The growth optimality assumption supposes that biological systems have evolved to maximize a given biological function. Typically, it is often considered that bacteria aim at maximizing their growth rate or biomass production (Feist and Palsson, 2010). Let 'growth' $\in \mathcal{R}$ be a reaction modeling the biomass production. The FBA assumes that the flux v_{growth} is maximized, *i.e.* maximize $\quad v_{\text{growth}}$.

Definition. Formally, the FBA is formulated as a linear optimization problem.

▶ **Definition 1.2: Flux Balance Analysis (Orth et al., 2010)**

Given a metabolic network $\mathcal{N} = (\mathcal{M}_{\text{ext}} \cup \mathcal{M}_{\text{int}}, \mathcal{R}, S)$, and $l_r, u_r \in \mathbb{R}$ flux bounds for each reaction $r \in \mathcal{R}$, the Flux Balance Analysis (FBA) is defined as:

$$\begin{aligned}
&\text{maximize} && v_{\text{growth}} && && \text{(I.1)}\\
&\text{such that} && S_{\mathcal{M}_{\text{int}},\mathcal{R}} \cdot v = 0 && && \text{(I.2)}\\
&\text{and} && l_r \leq v_r \leq u_r && \forall\, r \in \mathcal{R} && \text{(I.3)}\\
&\text{with} && v_r \in \mathbb{R} && \forall\, r \in \mathcal{R} &&
\end{aligned}$$

where $S_{\mathcal{M}_{\text{int}},\mathcal{R}}$ is the submatrix of S whose rows correspond to internal metabolites.

For the rest, a flux distribution $v \in \mathbb{R}^{|\mathcal{R}|}$ is a *metabolic steady-state* (MSS) if it satisfies Eqs. I.2 and I.3. The set of all metabolic steady-states compatible with a metabolic network $\mathcal{N}$ is denoted by $\text{MSS}(\mathcal{N})$.

Example. For instance, let us consider the metabolic network shown in Fig. 3. Each MSS $v \in \mathbb{R}^9$ satisfies the following linear constraints. The steady-state constraints (Eq. I.2) are:

$$\begin{aligned}
\frac{d[\text{A}]}{dt} = 0 &\iff 1 \times v_{\text{Tc1}} + 1 \times v_{\text{Tc2}} - 1 \times v_{\text{Growth}} - 1 \times v_{\text{R6}} - 1 \times v_{\text{R7}} = 0\\
\frac{d[\text{D}]}{dt} = 0 &\iff 3 \times v_{\text{R6}} - 1 \times v_{\text{Td}} = 0\\
\frac{d[\text{E}]}{dt} = 0 &\iff 3 \times v_{\text{R7}} - 1 \times v_{\text{Te}} = 0\\
\frac{d[\text{O2}]}{dt} = 0 &\iff 1 \times v_{\text{To2}} - 1 \times v_{\text{Rres}} = 0\\
\frac{d[\text{ATP}]}{dt} = 0 &\iff 1 \times v_{\text{Rres}} + 2 \times v_{\text{R6}} - 1 \times v_{\text{Growth}} = 0\\
\frac{d[\text{NADH}]}{dt} = 0 &\iff 1 \times v_{\text{Growth}} - 1 \times v_{\text{Rres}} - 3 \times v_{\text{R7}} = 0
\end{aligned}$$

Given the flux bounds: $(l_{\text{Tc1}}, u_{\text{Tc1}}) = (l_{\text{Tc2}}, u_{\text{Tc2}}) = (0, 10.5)$, $(l_{\text{Td}}, u_{\text{Td}}) = (l_{\text{Te}}, u_{\text{Te}}) = (0, 12.0)$, $(l_{\text{R6}}, u_{\text{R6}}) = (l_{\text{R7}}, u_{\text{R7}}) = (l_{\text{Rres}}, u_{\text{Rres}}) = (l_{\text{Growth}}, u_{\text{Growth}}) =$

$(0, 9999)$ and $(l_{\text{To2}}, u_{\text{To2}}) = (0, 15.0)$, the flux bounds constraints (Eq. I.3) are:

$$0 \leq v_{\text{Tc1}} \leq 10.5 \quad 0 \leq v_{\text{Tc2}} \leq 10.5 \quad 0 \leq v_{\text{Td}} \leq 12.0 \quad 0 \leq v_{\text{Te}} \leq 12.0$$
$$0 \leq v_{\text{To2}} \leq 15.0 \quad 0 \leq v_{\text{R6}} \leq 9999.0 \quad 0 \leq v_{\text{R7}} \leq 9999.0 \quad 0 \leq v_{\text{Rres}} \leq 9999.0$$
$$0 \leq v_{\text{Growth}} \leq 9999.0$$

The optimal MSSs computed by the FBA are the MSSs that maximize the flux v_{Growth} through the reaction 'Growth'.

Dynamic Flux Balance Analysis. The *dynamic* FBA (dFBA) is an extension of FBA allowing the simulation of a metabolic network over time (Mahadevan et al., 2002). There are two formulations of dFBA: a dynamic optimization approach (DOA), and a static optimization approach (SOA). For both formulations, the simulation is divided into timesteps of fixed length. It is assumed that the metabolism activity is constant during each timestep. The concentrations of environmental metabolites are updated between each timestep to account for the production and consumption of environmental metabolites during the previous timestep. For DOA, a linear optimization problem that extends the FBA equations is solved. It introduced new linear constraints to constrain metabolic flux changes between each timestep. For SOA, the FBA equations are solved successively and independently for each timestep, updating only the exchange reaction bounds between timesteps. An example of dFBA (SOA) simulation of the metabolic network of Fig. 3 is shown in Fig. 4.

Ressource Balance Analysis. The FBA framework is not the only way to model the metabolism activity through flux distributions. One of them is the *ressource balance analysis* (RBA) framework (Goelzer and Fromion, 2011) that incorporates additional constraints related to the availability and allocation of cellular resources (*e.g.* enzymes, proteins). The RBA framework assumes that biological systems optimize the use of their resources: the system must be the most efficient possible with a limited set of resources. To do that, RBA incorporates the cost of producing and recycling cellular resources to compute flux distributions that minimize resource consumption while maximizing growth. By minimizing resource consumption, RBA models can reproduce complex behaviors associated with the regulatory system (*e.g.* diauxic shift) (Tournier et al., 2017). The dRBA formalism is a dynamic extension of RBA (Jeanne et al., 2018).

In this manuscript, we focus on FBA-based frameworks rather than RBA-based frameworks. It necessitates fewer parameters to reconstruct FBA-compatible metabolic networks than RBA-compatible ones. Therefore, almost all metabolic networks available are FBA-compatible, and not RBA-compatible.

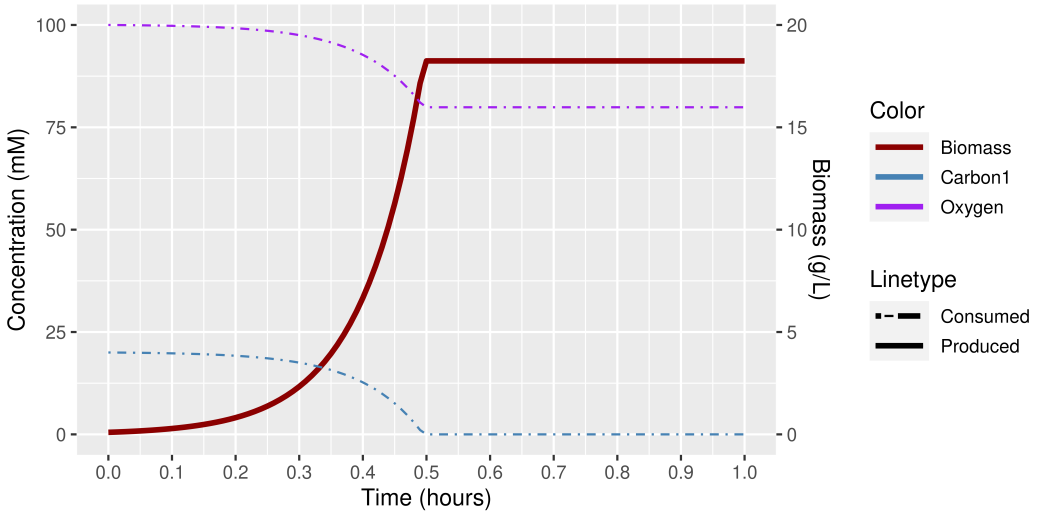


Figure 4 – Dynamic FBA (SOA) simulation of the *toy* metabolic network (Fig. 3) made with FlexFlux, The thick red line is the biomass concentration, and the dotted lines are the environmental metabolite concentrations. The simulation has been made with a timestep duration $\tau = 0.01$h and substrate concentrations initialized at 100mM for Oxygen, at 20mM for Carbon1, and 0mM for Carbon2. The flux bounds are $\forall r \in \{\text{Tc1}, \text{Tc2}\}$, $(l_r, u_r) = (0, 10.5)$, $\forall r \in \{\text{Td}, \text{Te}\}$, $(l_r, u_r) = (0, 12.0)$, $\forall r \in \{\text{R6}, \text{R7}, \text{Rres}, \text{Growth}\}$, $(l_r, u_r) = (0, 9999)$ and $(l_{\text{To2}}, u_{\text{To2}}) = (0, 15.0)$.

1.2 Modeling the Activity of the Regulatory Scale

The regulatory scale encompasses all the mechanisms, or 'chemical rules', that control gene expression and cellular activities through the action of regulatory proteins, also called transcription factors, and signaling pathways. Regulatory processes can span a broader range of timescales than metabolic processes. Immediate post-translational modifications, such as phosphorylation, occur within seconds, while changes in gene expression can take minutes to hours.

For a gene to be expressed, it must be transcribed into messenger RNA (mRNA) that may be translated into proteins. Among these proteins are regulatory proteins, special proteins that influence gene expression, and enzymes, proteins that catalyze reactions. Gene expression is influenced by interactions between chemical components, including DNA-protein and protein-protein interactions.

For five decades now, methods have been developed to model, simulate, and infer gene regulatory networks and their Boolean dynamics (de Jong, 2002; Bernot et al., 2004; Chaves et al., 2010).

1.2.1 Gene Regulatory Networks

The interactions between components of the regulatory scale are typically represented by gene regulatory networks (GRNs). A GRN is a static model that depicts the various influences (activation and inhibition) that genes and regulatory proteins exert on one another, especially the influence impacting gene expression. GRNs do not capture the ways influences cooperate or compete to regulate gene expression.

Figure 5 shows the GRN that we generated from the description of *Escherichia coli* core metabolism, whose metabolic network is shown in Fig. 2. It contains 235 interactions of which 60 are protein-protein or DNA-protein interactions.

Inference of interactions. Interactions are generally inferred from gene expression data using statistical methods, inferred from experimental observations of proteins binding to gene promoters, extracted from the literature, or manually curated (Badia-i Mompel et al., 2023). The quality of these interactions strongly depends on the method used to infer them. Interactions statistically inferred or deduced from experimental observation are not necessarily true. It is not because a protein is seen binding to a gene promoter, or correlated to a gene expression, that the protein has a causal influence on the gene expression. Therefore, apart from manually curated interactions, the interactions included in GRNs should be considered potential sources of influence on gene expression and must be carefully selected.

Databases. There are many databases of interactions, such as *RegulonDB* (Salgado et al., 2023), which is specialized for *Escherichia coli*, and *CollecTRI* (Müller-Dott et al., 2023), a recently published database of known interactions in the regulatory scale of humans and mice. Most interactions in these databases are not manually curated. Additionally, no database currently integrates the interactions between components of the metabolic and regulatory scales.

In this manuscript, we do not rely on the interactions available in these databases to generate our search spaces and explain gene activities. Instead, we rely solely on interactions already validated by experts or already used in models combining the metabolic and regulatory scales.

1.2.2 Boolean Networks

Boolean networks (BN) are a well-established approach to model the dynamics of the regulatory scale (Kauffman, 1969; Thomas, 1973; Wang et al., 2012). It assumes that each gene and regulatory protein is either *active* (1) or *inactive* (0), *i.e.* expressed or not. Given $\mathbb{B} = \{0; 1\}$ the Boolean domain, the regulatory state

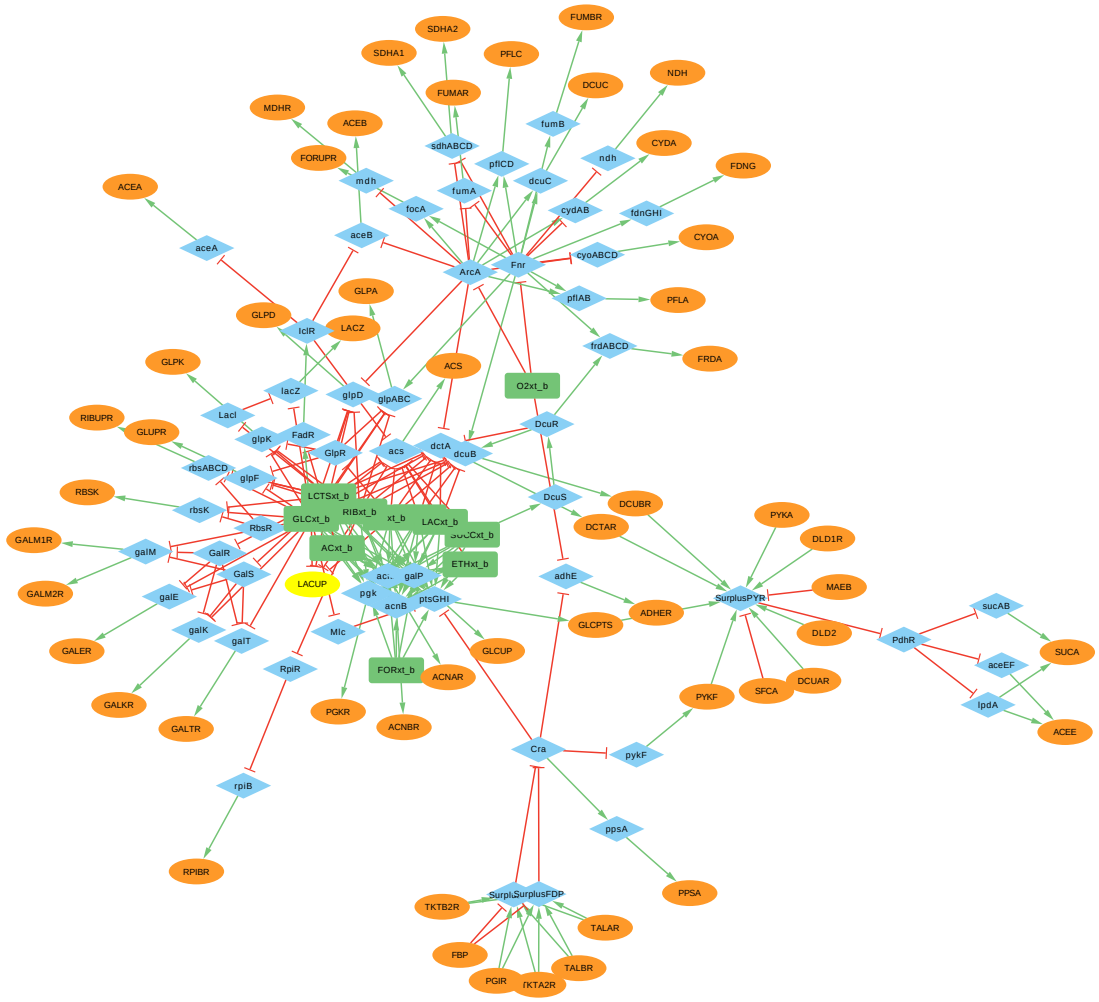


Figure 5 – Gene regulatory network associated with the core metabolism of *Escherichia coli* (Fig. 2), generated from the description provided in Covert and Palsson (2002). Only interactions from and to components that are regulated, *i.e.* not always expressed, are shown. Blue nodes are genes that encode for enzymes and regulatory proteins, green nodes are environmental metabolites, and orange nodes are reactions. Green arrows are positive interactions (activation effect), and red arrows are negative interactions (inhibition effect).

a	b	$\neg a$	$a \wedge b$	$a \vee b$
0	**0**	1	0	0
0	**1**	1	0	1
1	**0**	0	0	1
1	**1**	0	1	1

Table 1 – Truth tables for the logical '*not*' ('$\neg$'), the logical '*and*' ('$\wedge$'), and the logical '*or*' ('$\vee$') operators. $a, b \in \mathbb{B} = \{0; 1\}$ are two Boolean-valued variables. The true value is 1, and the false value is 0.

is a Boolean vector $x \in \mathbb{B}^n$ that associates a state (active or inactive) to n genes and regulatory proteins.

Biologically, the state of a component is influenced by all the components interacting with it. The activation or inhibition of a component depends on the cooperation and the competition between the influences it receives from other components. In practice, the activation condition is expressed by a Boolean logic function built from the logical '*not*' ('$\neg$'), logical '*and*' ('$\wedge$'), and logical '*or*' ('$\vee$') operators, and Boolean-valued variables. The truth tables for these three logical operators are recalled in Table. 1. For example, the Boolean logic function $(a \vee b) \wedge c$ over the three Boolean-valued variables $a, b, c \in \mathbb{B} = \{0; 1\}$ is true whenever $c = 1$, and either $a = 1$ or $b = 1$.

Boolean networks. A Boolean network (BN) is a set of Boolean logic functions that describe the activation states of each regulatory component. Formally, BNs are commonly defined as follows.

▶ **Definition 1.3: Boolean network**

A *Boolean network* (BN) of dimension n is a function

$$f : \mathbb{B}^n \to \mathbb{B}^n$$

For each $i \in \{1, \cdots, n\}$, the i-th component $f_i : \mathbb{B}^n \to \mathbb{B}$ is called the *local function of* i.

Influence Graphs. Each local function $f_i : \mathbb{B}^n \to \mathbb{B}$ of a BN f is based on a set of influences between components. A component j has an influence on f_i if and only if it exists $x \in \mathbb{B}^n$ such that changing the j-th value of x change the output of f_i. The influence is positive (*resp.* negative) if increasing j-th value can increase (*resp.* decrease) the output of f_i.

The set of all influences of a BN is summarized into an influence graph, also called an interaction graph. An *influence graph* is a signed digraph such that positive edges model positive influences, while negative edges model negative influences. Influence graphs are commonly used to identify properties of BN dynamics, *e.g.* ensuring the existence of steady-states from negative and positive loops in the influence graph (Thomas, 1981). The influence graph of a BN modeling a regulatory system is a gene regulatory network (GRN).

▶ **Definition 1.4: Influence graph**

Given a Boolean network f of dimension n, the *influence graph* of f is a signed digraph

$$G(f) = (V, E)$$

with $V = \{1, \cdots n\}$ and $E \subseteq V \times \{-, +\} \times V$ such that $(j, s, i) \in E$ if and only if there exists $x \in \mathbb{B}^n$ such that:

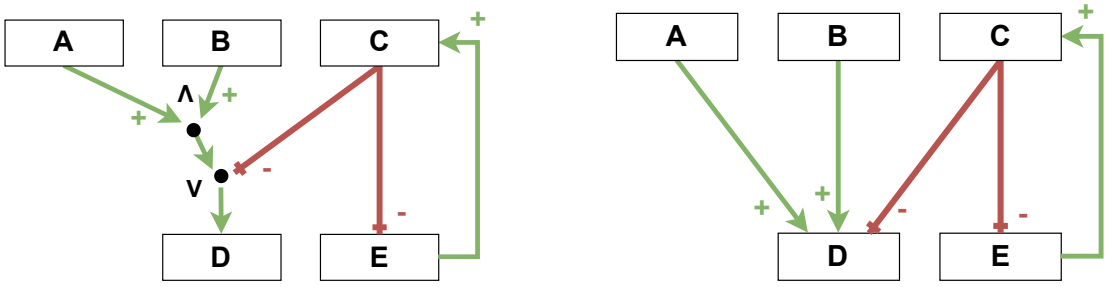
$$s \cdot f_i(x_1, \cdots, x_{j-1}, 0, x_{j+1}, \cdots, x_n) < s \cdot f_i(x_1, \cdots, x_{j-1}, 1, x_{j+1}, \cdots, x_n)$$

A BN f is *locally monotone* whenever for each influence $(j, s, i) \in G(f)$, there is no influence with the opposite sign, *i.e.* $(j, -s, i) \notin G(f)$.

Example. An example of Boolean network $f : \mathbb{B}^5 \to \mathbb{B}^5$ of dimension $n = 5$ is provided in Fig. 6a. It is composed of 5 local functions:

$$f_A(x_A, x_B, x_C, x_D, x_E) = 1 \qquad f_B(x_A, x_B, x_C, x_D, x_E) = 1$$
$$f_C(x_A, x_B, x_C, x_D, x_E) = x_E \qquad f_D(x_A, x_B, x_C, x_D, x_E) = (x_A \wedge x_B) \vee \neg x_C$$
$$f_E(x_A, x_B, x_C, x_D, x_E) = \neg x_C$$

The local functions of A and B are constants (always true). The influence graph of this Boolean network is shown in Fig. 6b. It is composed of 5 influences $(\{(A, +, D), (B, +, D), (C, -, D), (C, -, E), (E, +, C)\})$ over the nodes $\{A, B, C, D, E\}$. For instance, the local function $f_D : \mathbb{B}^5 \to \mathbb{B}$ is a composition with a logical '*and*' ($\wedge$) and logical '*or*' ($\vee$) operators of 3 influences: 2 positive influences from A and B ($(A, +, D)$ and $(B, +, D)$), and 1 negative influence from C $(C, -, D)$. Indeed, we have $f_D(0, 1, 1, 0, 0) < f_D(1, 1, 1, 0, 0)$, $f_D(1, 0, 1, 0, 0) < f_D(1, 1, 1, 0, 0)$, and $f_D(0, 0, 0, 0, 0) > f_D(0, 0, 1, 0, 0)$.



(a) Boolean Network (BN) **(b)** Influence graph

Figure 6 – Example of a Boolean network $f : \mathbb{B}^5 \to \mathbb{B}^5$ of dimension $n = 5$ (a) and of its influence graph (b). Green arrows are positive influence, modeling activation effects, and red arrows are negative influence, modeling inhibition effects.

1.2.3 Semantics of Boolean Networks Dynamics

Given a Boolean network (BN) $f : \mathbb{B}^n \to \mathbb{B}^n$ of dimension n, a *configuration* of the BN is a Boolean vector $x \in \mathbb{B}^n$. For regulatory systems, the configuration is called a *regulatory state*. Different semantics of BN update modes can be used to determine how configurations, or regulatory states, evolved. The most commonly used semantics are the *synchronous* update (Kauffman, 1969) and the *asynchronous* update (Thomas, 1973).

Synchronous update. With the synchronous update, all the components are updated simultaneously to compute the next configuration $x' \in \mathbb{B}^n$, *i.e.* $x' = f(x)$. This semantics is deterministic, there is only one transition possible from any configuration. Consider the Boolean network $f : \mathbb{B}^5 \to \mathbb{B}^5$ described in Fig. 6a with the configuration $\{x_A = 0, x_B = 1, x_C = 1, x_D = 1, x_E = 0\}$, the next configuration according to the synchronous update semantic is $x' = \{x'_A = 1, x'_B = 1, x'_C = 0, x'_D = 0, x'_E = 0\}$.

Asynchronous update. The standard asynchronous update consists of updating only one component i at a time. The next configuration $x' \in \mathbb{B}^n$ is such that $\forall\, 0 \leq j \leq n, j \neq i,\ x'_j = x_j$ and $x'_i = f_i(x)$. There are at most 2^n configurations that can be generated from any configuration. With the Boolean network $f : \mathbb{B}^5 \to \mathbb{B}^5$ described in Fig. 6a of previous example, the configurations $x^1 = \{x^1_A = 1, x^1_B = 1, x^1_C = 1, x^1_D = 1, x^1_E = 0\}$ and $x^2 = \{x^2_A = 0, x^2_B = 1, x^2_C = 1, x^2_D = 0, x^2_E = 0\}$ are two reachable configurations from $\{x_A = 0, x_B = 1, x_C = 1, x_D = 1, x_E = 0\}$ according to an asynchronous update semantics.

In practice, the asynchronous semantics is often considered more biologically relevant than the synchronous semantics. The synchronous semantics assumes

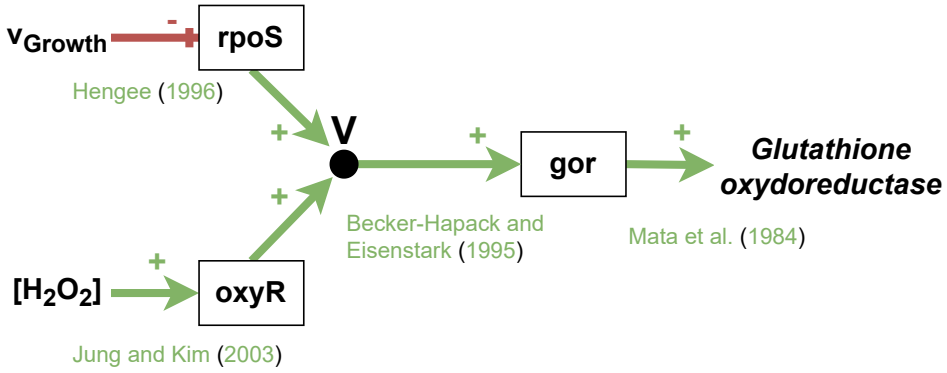


Figure 7 – Example of interconnection between the regulatory and metabolic scales extracted from the regulatory network of *Escherichia coli* introduced in Covert et al. (2004). The activity of the '*Growth*' reaction and the availability of H_2O_2 in the cell environment is needed to transcribe the genes *rpoS* and *oxyR*, respectively. The gene *gor* should be expressed to produce the enzyme catalyzing the reaction of *glutathione oxydoreductase*.

that all regulatory processes occur at the same speed, which is not how regulatory processes work[1]. Despite that, most of the simulation frameworks based on FBA that handle regulatory rules assume a synchronous update semantic (see Section 1.3.2). Therefore, in this manuscript, we consider BNs with a synchronous update semantic.

1.3 Coupling the Regulatory and Metabolic Scales

While mostly modeled and studied separately, the regulatory and metabolic scales are in reality interconnected (Covert et al., 2001; Oyarzún et al., 2012; Zañudo et al., 2017; Chaves et al., 2019; Carthew, 2021). The production of the enzymes catalyzing the reactions is controlled by a cascade of regulatory mechanisms involving regulatory proteins, metabolites, and abiotic factors (*e.g.* temperature, pH). Additionally, some metabolic byproducts bind to regulatory proteins or inhibit/activate gene expression, thereby indirectly influencing the set of reactions occurring within the metabolism. Consequently, the metabolism has a *feedback* effect on the regulatory system, which in turn *controls* the metabolic activity.

[1] While considered more accurate than synchronous semantics, asynchronous semantics do not model all regulatory behaviors. More complex update semantics, such as the *Most Permissive* semantics (Paulevé et al., 2020), are needed for that.

Examples. Figure 7 show the cascade of 4 regulatory rules controling the reaction of *glutathione oxydoreductase* in *Escherichia coli*'s metabolism. The catalysis of this reaction is indirectly dependent on the availability of H_2O_2 in the cellular environment and of the cell growth (modeled by the flux v_{Growth}). This external metabolite and this reaction impact the transcription of the genes *oxyR* and *rpoS*, respectively, which enables the expression of the gene *gor*, whose transcription produces the enzyme catalyzing for the *glutathione oxydoreductase*.

The interaction between the two scales can also be seen in the gene regulatory network (GRN) of *Escherichia coli*'s core metabolism in Fig. 5. Among the interactions described in the GRN, there are 48 gene-reaction interactions and 109 metabolite-gene interactions.

1.3.1 Regulated Metabolic Networks

The multilayered structure of biological systems can be modeled using regulated metabolic networks (Covert et al., 2001; Oyarzún et al., 2012; Marmiesse et al., 2015; Chaves et al., 2019). A *regulated metabolic network* consists of two interdependent networks: **(i)** the regulatory network, modeled by a Boolean network (BN); and **(ii)** the metabolic network. The regulatory network can control the activity of the metabolic network's reactions by forcing inhibited reactions to have a zero flux (*control rules*). It takes as input components of the metabolic network, including reaction states (active or not) and environmental metabolite availabilities (*feedback rules*). From the regulatory network point of view, a reaction is active if it has a non-null metabolic flux, and a metabolite is available if it has a non-null concentration in the cell medium.

There is no formal definition for regulated metabolic networks in the literature. Thus, we introduce a formal definition in Thuillier et al. (2021) (Chapter III) that we use throughout all our works.

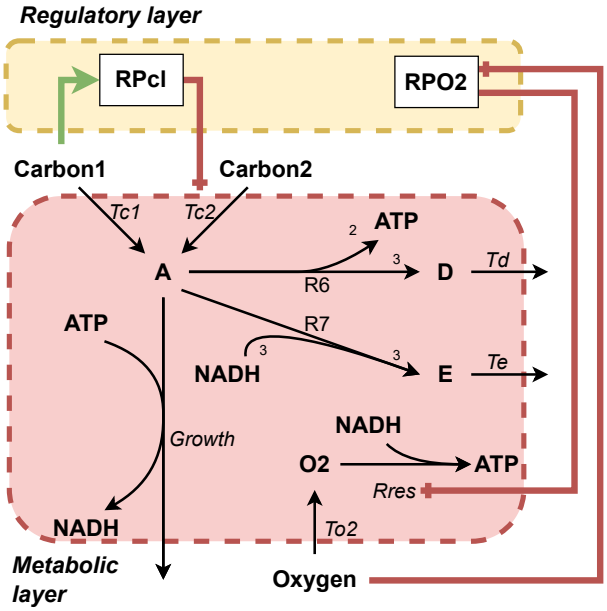
▶ **Definition 1.5: Regulated metabolic network (Thuillier et al., 2021)**

A *regulated metabolic network* is a triplet $(\mathcal{N}, \mathcal{P}, f)$ composed of:

— a metabolic network $\mathcal{N} = (\mathcal{M} = \mathcal{M}_{\text{ext}} \cup \mathcal{M}_{\text{int}}, \mathcal{R}, s)$ with $k = |\mathcal{M}_{\text{ext}}|$ external metabolites, and $m = |\mathcal{R}|$ reactions;

— a set of d genes and regulatory proteins $\mathcal{P}$;

— a BN f of dimension $n = k + m + d$ where $\{1, \cdots, n\} = \mathcal{M}_{\text{ext}} \cup \mathcal{R} \cup \mathcal{P}$ such that the interaction graph $G(f)$ of f is a bipartite graph between $\mathcal{P}$ and $\mathcal{M}_{\text{ext}} \cup \mathcal{R}$.



Boolean network:

$$f_{\mathrm{RPcl}}(x) = [\mathrm{Carbon1}] > 0$$
$$f_{\mathrm{RPO2}}(x) = \neg([\mathrm{Oxygen}] > 0)$$
$$f_{\mathrm{Tc2}}(x) = \neg x_{\mathrm{RPcl}}$$
$$f_{\mathrm{Rres}}(x) = \neg x_{\mathrm{RPO2}}$$

Figure 8 – Example of a *toy* regulated metabolic network introduced in Thuillier et al. (2021). It is a simplified model of the model of core-carbon metabolism introduced in Covert et al. (2001). Square nodes are regulatory proteins, and colored arrows are influences between components: the green arrow is a positive influence, and the red arrows are negative influences. The Boolean logical functions associated with the regulatory network are on the right side of the figure. Only non-constant functions are shown. The metabolic network (red square) is the same as in Fig. 3.

Toy example. An example of a *toy* regulated metabolic network is shown in Fig. 8. This example extends the metabolic network shown in Fig. 3 with Boolean regulatory rules. It is a simplified model of core-carbon metabolism, originally proposed in Covert et al. (2001). At the metabolic level, there are $m = 9$ reactions and $k = 3$ input metabolites.

At the regulatory level, there are $d = 2$ regulatory proteins: $\mathcal{P} = \{\mathrm{RPcl}, \mathrm{RPO2}\}$. Thus, the Boolean network f is of dimension $n = k + m + d = 14$. It consists of 14 functions which map a Boolean vector $x = \{x_{\mathrm{Carbon1}}, x_{\mathrm{Carbon2}}, x_{\mathrm{Oxygen}}, x_{\mathrm{RPcl}}, x_{\mathrm{RPO2}}, x_{\mathrm{Tc1}}, x_{\mathrm{Tc2}}, x_{\mathrm{To2}}, x_{\mathrm{Td}}, x_{\mathrm{Te}}, x_{\mathrm{Growth}}, x_{\mathrm{Rres}}, x_{\mathrm{R6}}, x_{\mathrm{R7}}\} \in \mathbb{B}^n$ to a Boolean value in $\mathbb{B}$. The local functions associated with regulatory proteins in $\mathcal{P}$ involve only external metabolite variables. Among the 9 functions associated with reactions, only two (Tc2, Rres) are non-constant: they involve the two regulatory proteins. The 3 functions associated with environmental metabolites are considered constant. The regulatory state of metabolites is dependent on their concentration in the medium, $\forall\, m \in \{\mathrm{Carbon1}, \mathrm{Carbon2}, \mathrm{Oxygen}\}$, $f_m(x) = [m] > 0$.

Model	Metabolic layer # reactions	Cycle?	Regulatory layer Dimension	# controls	# feedbacks
Toy	9		4	2	2
Core	20	✓	11	7	4
Medium-scale	113	✓	151	90	35

Table 2 – Summary of the size and complexity of the three regulated metabolic networks used in this manuscript: *Toy* (Thuillier et al., 2021), *Core* (Covert et al., 2001), and *Medium-scale* (Covert and Palsson, 2002). '#controls' and '#feedback' denote the number of control and feedback regulatory rules, respectively.

RMNs used in this manuscript. The methods presented in this manuscript have been validated and benchmarked using three regulated metabolic networks of *Escherichia coli* of increasing complexity: **(i)** the regulated metabolic network presented in Fig. 8, referred to as the *toy* model; **(ii)** the model of core-carbon metabolism (Covert et al., 2001), referred to as the *core* model; and **(iii)** a *medium-scale* model whose metabolic network is shown in Fig. 5 (Covert and Palsson, 2002), referred to as the *medium-scale* model.

At the metabolic level, the difference in complexity relies on the size and the structure of the metabolic networks. The *toy* model has 9 reactions and no metabolic cycles, while the *core* and *medium-scale* models have 20 and 113 reactions, respectively, with metabolic cycles.

At the regulatory level, the difference in complexity relies on the regulatory network dimension, the number of control and feedback rules, and the regulatory rules structures. The *toy* model has 4 regulatory rules, of which 2 are controls rules and 2 are feedback rules. The *core* model has 11 regulatory rules, of which 7 are control rules and 4 are feedback rules. For both the *toy* and *core* models, regulatory rules are "simple", *i.e.* they are influenced by only one component. In contrast, the *medium-scale* model introduces "complex" regulatory rules, *i.e.* regulatory rules that are compositions of several influences. It has 151 non-constant regulatory rules, of which 90 are control rules and 35 are feedback rules. The complexity difference of these three models is summarized in Table 2.

The *toy* model is described in chapter III, and the *core* and *medium-scale* models are described in Appendix A.

1.3.2 Regulatory Flux Balance Analysis

To figure out how gene expression triggers specific phenotypes depending on the environmental constraints (Buescher et al., 2012), several constraint-based approaches have been developed to integrate the metabolic and regulatory scales activities (Liu and Bockmayr, 2020; Moulin et al., 2021). These approaches are

Approaches	Without resource costs	With resource costs
Static	SR-FBA (Shlomi et al., 2007), PROM (Chandrasekaran and Price, 2010)	-
Iterative	rFBA (Covert et al., 2001), iFBA (Covert et al., 2008)	idFBA (Min Lee et al., 2008)
Dynamics	-	r-deFBA (Liu and Bockmayr, 2020)

Table 3 – Summary of the state-of-the-art constraint-based flux balance formalisms (with and without considering resource costs) that handle regulations. This table is adapted from table 1 of Liu and Bockmayr (2020).

summarized in Table 3. Except for iFBA, which employs ordinary differential equations (ODE) to model the regulatory scale activity, and r-deFBA, which models the metabolic fluxes with ODE equations, all other approaches integrate the FBA with a Boolean network. These approaches mostly rely on synchronous update semantics, except PROM which relies a probabilistic update semantics.

The most comprehensive formalisms are dynamic formalisms, which consider both the cost of enzymes and regulation (de-rFBA). However, they are also the most complex abstractions, requiring extensive parameter estimation and calibration.

rFBA. In this manuscript, we rely on the *regulatory flux balance analysis* (rFBA) formalism (Covert et al., 2001), an extension of FBA that integrates dFBA (SOA) with the synchronous dynamics of a Boolean network. To couple the different timescales of the regulatory and metabolic systems, rFBA divides the simulation into fixed-length timesteps. As metabolic processes are much faster than regulatory processes (occurring in seconds versus minutes to hours), the metabolic state is assumed stable and constant throughout each timestep. The state of the regulatory network is updated once between each timestep. At the end of each timestep, the concentrations of extracellular metabolites are updated based on the metabolic fluxes of exchange reactions. The functions for updating these concentrations and the biomass are the same as for dFBA (SOA) (Varma and Palsson, 1994).
We introduce a formal definition of rFBA in the paper Thuillier et al. (2021), described in Chapter III.

With rFBA, each timestep is composed of **(i)** a metabolic state that associates a flux value to each reaction; **(ii)** a substrate state that associates a concentration to each external metabolite; and **(iii)** a regulatory state that associates a Boolean state to each regulated or regulating component. In this manuscript, we denote the global states of the regulated metabolic networks at each timestep as regulated metabolic steady-states (RMSS).

▶ **Definition 1.6: Regulated metabolic steady-state (Thuillier et al., 2021)**

A *regulated metabolic steady-state* (RMSS) is a triplet

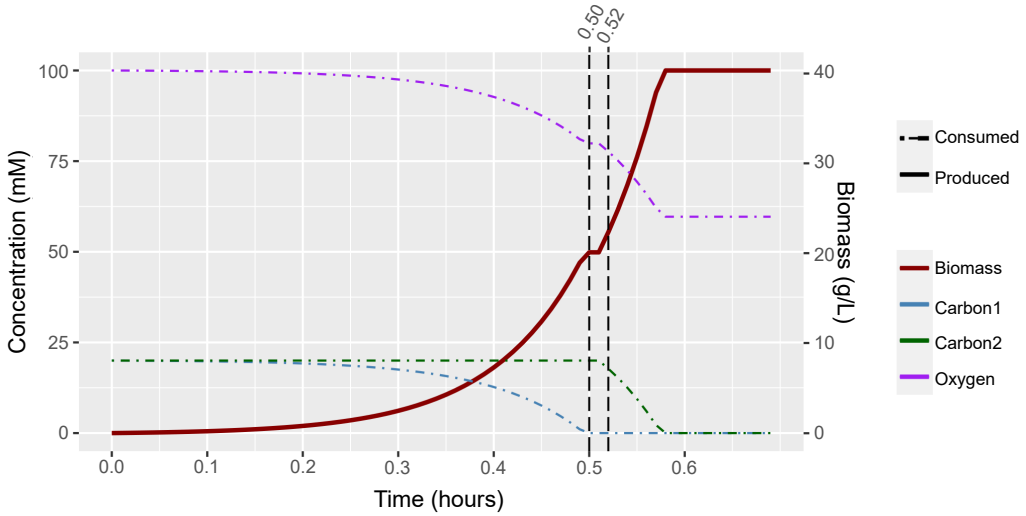
$$(v, w, x) \in \mathbb{R}^{|\mathcal{R}|} \times \mathbb{R}^{|\mathcal{M}_{\text{ext}}|} \times \mathbb{B}^{|\mathcal{M}_{\text{ext}}|+|\mathcal{R}|+|\mathcal{P}|}$$

where $v \in \mathbb{R}^{|\mathcal{R}|}$ is a metabolic steady-state (satisfying Eqs. I.2 and I.3) in which reactions inhibited by the regulatory state $x \in \mathbb{B}^{|\mathcal{M}_{\text{ext}}|+|\mathcal{R}|+|\mathcal{P}|}$ have a null metabolic fluxes, *i.e.* $\forall r \in \mathcal{R}, x_r = 0 \implies v_r = 0$. The vector $w \in \mathbb{R}^{|\mathcal{M}_{\text{ext}}|}$ represents the concentrations of extracellular metabolites. It is used to compute the flux bounds l_r and u_r of exchange reactions.

For the rest, we will denote by $\text{rMSS}(\mathcal{N}, w, x)$ the set of all metabolic steady-states $v \in \text{MSS}(\mathcal{N})$ such that (v, w, x) is an RMSS.

Example. An rFBA simulation of the *toy* regulated metabolic network (Fig. 8) is shown in Fig. 9a. The simulation models a diauxic shift, a sequential consumption of two substrates when both are initially available. This mechanism was first demonstrated in 1942 by Monod (1942), it is a well-known example of controls exerted by regulations on the metabolism. The simulation is performed with FlexFlux (Marmiesse et al., 2015), using a timestep of 0.01h and initial concentrations of 100 mM Oxygen, 20 mM Carbon1, and 20 mM Carbon2. The simulation contains 70 timesteps.

More precisely, the simulation shows that until 0.5h only Carbon1 and Oxygen are consumed to produce biomass. This corresponds to a first growth phase where the system's behavior is constant. The presence of Carbon1 activates the regulatory protein RPcl inhibiting the reaction Tc2 according to the regulatory rules. At 0.5h, Carbon1 is depleted, and the current Boolean state $x \in \mathbb{B}^{14}$ is such that $x_{\text{Carbon1}} = 0$, $x_{\text{RPcl}} = 1$, $x_{\text{Tc2}} = 0$. At 0.51h, as shown in Fig. 9b, the Boolean state x is updated to x' so that the Boolean state of RPcl becomes $x'_{\text{RPcl}} = f_{\text{RPcl}}(x) = x_{\text{Carbon1}} = 0$. The Boolean state of Tc2 remains unchanged because $x_{\text{RPcl}} = 1$. No biomass is produced at 0.51h. At 0.52h, the Boolean state x' is updated to x'': all the node states remain unchanged except for $x''_{\text{Tc2}} = f_{\text{Tc2}}(x') = \neg x'_{\text{RPcl}} = 1$. The reaction Tc2 is not inhibited anymore, and the biomass is produced due to the uptake of Carbon2 and Oxygen (through Growth, Tc2, and Rres) until Carbon2 depletion at 0.59h. It is the second growth phase.



(a) rFBA simulation made with FlexFlux, with a timestep duration $\tau = 0.01$h and substrate concentrations initialized at 100mM for Oxygen, and at 20mM for Carbon1 and Carbon2. The flux bounds are $\forall r \in \{\text{Tc1, Tc2}\}$, $(l_r, u_r) = (0, 10.5)$, $\forall r \in \{\text{Td, Te}\}$, $(l_r, u_r) = (0, 12.0)$, $\forall r \in \{\text{R6, R7, Rres, Growth}\}$, $(l_r, u_r) = (0, 9999)$ and $(l_{\text{To2}}, u_{\text{To2}}) = (0, 15.0)$.

Time	External metabolites			Regulatory proteins		Metabolic fluxes				
	w_{Carbon1}	w_{Carbon2}	w_{Oxygen}	x_{RPO2}	x_{RPcl}	v_{Tc1}	v_{Tc2}	v_{To2}	v_{Growth}	v_{Rres}
0.49h	2.95	20.0	82.95	0	1	10.5	0.0	10.5	10.5	10.5
0.50h	1.05	20.0	81.05	0	1	6.15	0.0	6.15	6.15	6.15
0.51h	0.0	20.0	79.90	0	0	0.0	0.0	0.0	0.0	0.0
0.52h	0.0	20.0	79.90	0	0	0.0	10.5	10.5	10.5	10.5
0.53h	0.0	17.76	77.65	0	0	0.0	10.5	10.5	10.5	10.5

(b) Focus on the substrate concentrations, regulatory protein states, and metabolic fluxes for five timesteps at Carbon1 depletion. The metabolic fluxes over the reactions Td, Te, R6, and R7 (not shown) are always equal to 0.

■ **Figure 9** – Dynamic rFBA simulation (a) of the regulated metabolic network of Fig. 8. (b) highlights five timesteps from 0.49h to 0.53h, at the carbon source shift (*i.e.* depletion of Carbon1). The simulation has been made with the tool FlexFlux.

2 Inference of Regulated Metabolic Networks

Up to now, very few approaches exploited the metabolic scale to infer regulatory information about the regulatory scale. In Tournier et al. (2017), resource balance analysis (RBA) (Goelzer and Fromion, 2011) is employed to manually infer logical rules that control the activation of metabolic fluxes in response to diverse extracellular media. However, this method assumes no feedback from metabolism to regulation, which does not correspond to the biological functioning of cells in most cases. Typically, these control and feedback rules are manually curated from literature or experimental data, as seen for models of *Escherichia coli* (Covert and Palsson, 2002; Covert et al., 2008) and a few other organisms like *Ralstonia solanacearum* (Peyraud et al., 2018). Therefore, the lack of automated frameworks for inferring Boolean rules interfacing the metabolism and regulatory system poses a significant limitation to the development of regulated metabolic models, and the use of accurate modeling formalisms.

Models are inferred separately. A regulated metabolic network is composed of two main components: a metabolic network and a Boolean network representing a regulatory network. Over the last decades, dedicated methods have been developed to reconstruct metabolic networks, and infer Boolean networks from *omics* data. Omics data provide comprehensive descriptions of cellular components and activity, including molecular component abundances, protein-protein and protein-DNA interactions, and functional-state of biological process (Joyce and Palsson, 2006). An overview of the omics data types is provided in Fig. 10.

In the next section, we review the state-of-the-art methods used to reconstruct metabolic networks from genomics, metabolomics, and fluxomics (Section 2.1) and infer Boolean networks from transcriptomics, proteomics, and interaction data (Section 2.2).

2.1 Inference of Metabolic Networks

The first genome-scale metabolic network (GMN) was reconstructed in 1999 for *Haemophilus influenzae* (Edwards and Palsson, 1999) using genome annotation methods, which map genes to enzymes that catalyze reactions. Since then, numerous high-quality GMNs have been reconstructed. As of 2021, over 6 000 GMNs have been recorded, with 75 species having high-quality, manually curated GMNs (King et al., 2015; Passi et al., 2021).

Reconstruction protocol. In 2010, *Thiele et al.* introduced a comprehensive protocol in 96 steps to reconstruct GMNs from genomic and experimental data, in particular

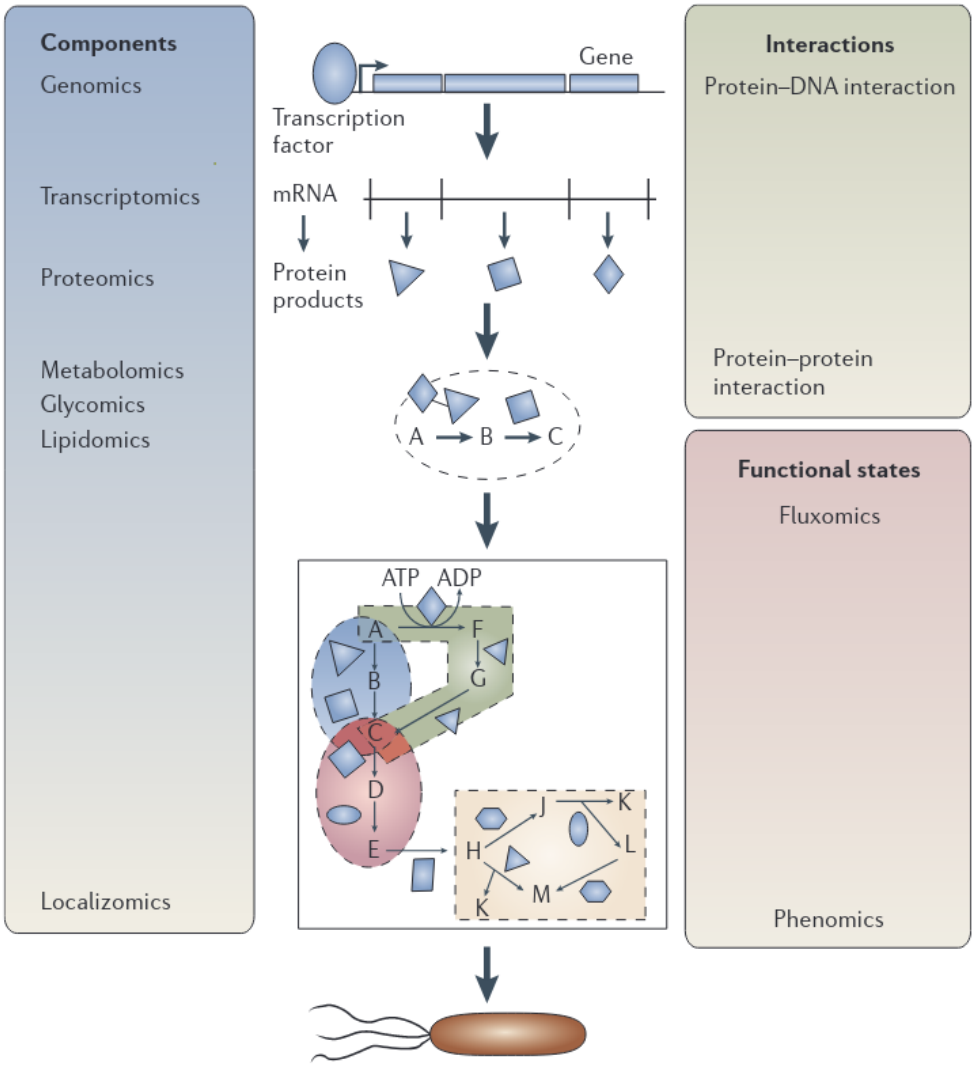


Figure 10 – Figure from Joyce and Palsson (2006): Overview of the omics data types and the element they quantify. The figure shows the sequence of biological processes and the type of omics data they are associated with. The DNA sequences (measured by *genomics* data) encoding for genes are transcribed into mRNA (*transcriptomics*) which may then be translated into proteins (*proteomics*), including enzymes. Enzymes are then used to catalyze reactions (*fluxomics*) that transform sets of metabolites into others (*metabolomics*). Protein-protein and protein-DNA interactions are associated with the regulatory processes.

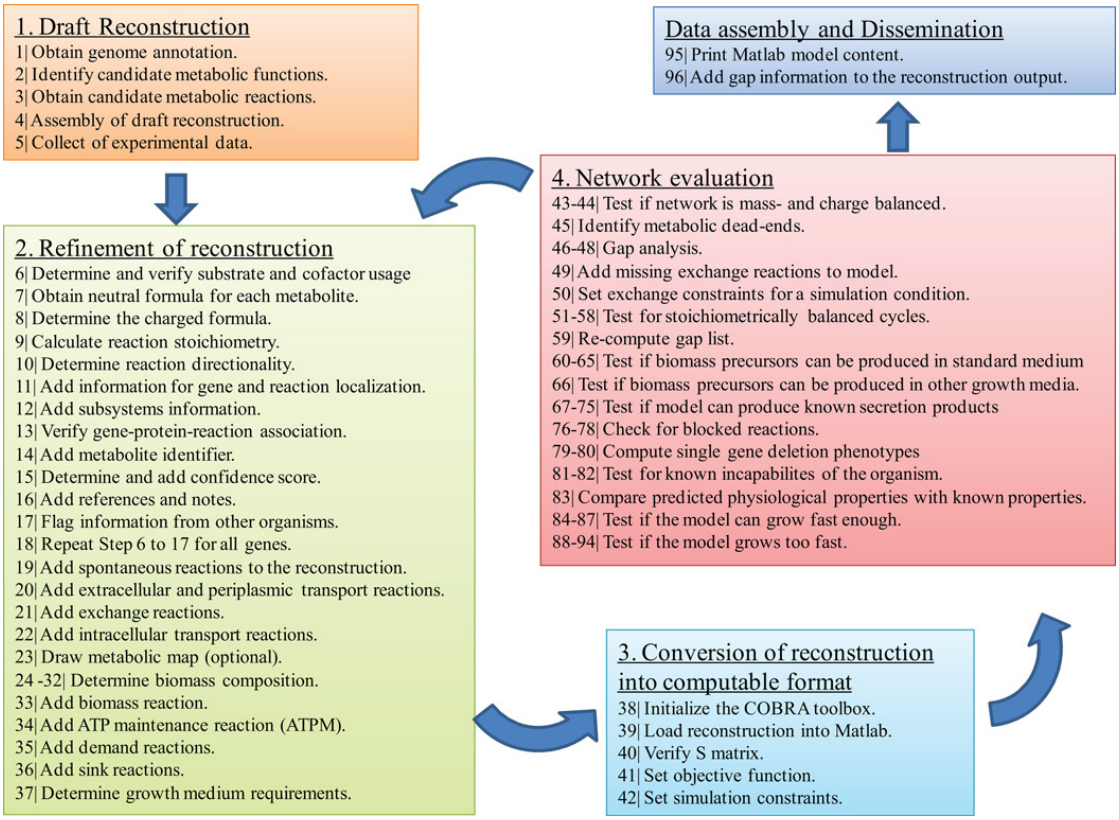


Figure 11 – Figure from Thiele and Palsson (2010): Overview of the 96 steps of the procedure to iteratively reconstruct metabolic networks.

metabolomics and fluxomics data (Thiele and Palsson, 2010). This protocol has been used to build some of the most highly curated GMNs, including GMNs of *Escherichia coli* (Orth et al., 2011). The 96 steps of the reconstruction protocol are shown in Fig. 11. They can be summarized into four main stages:

Stage 1: Draft reconstruction. A draft metabolic network is generated from genomics and proteomics data. The bacteria genes are identified and compared to databases of genes known to be involved in metabolic processes. For each of these genes, the associated reactions are retrieved and added to the metabolic network. This part of the reconstruction is the easiest to automate but is also prone to errors, such as missing reactions or spurious gene annotations.

Stage 2: Refinement. In this phase, the confidence of each reaction in the draft network is evaluated, and all low-confidence reactions are removed. The exchange reactions and the 'growth' reaction are defined during this step.

Stage 3: Conversion. The model is reduced to a stoichiometric matrix and a set

of reaction rate bounds. At this stage, FBA can be applied to the metabolic network.

Stage 4: Evaluation. The dynamic behavior of the metabolic network is compared with experimental observations (*e.g.* metabolomic or fluxomic data). Gap-filling approaches are used to infer missing reactions from the network topology to ensure all reactions are reachable and that a steady-state flux distribution compatible with the FBA exists. In particular, it is ensured that the growth reaction reflected exactly the experimental data; ensuring the system's growth rate is neither too fast nor too slow.

Stages 2, 3, and 4 are iterated until the metabolic network achieves a high confidence score. This refinement process is mostly manual and typically takes months to years.

GMNs are an input of our methods. The problem of metabolic network inference is not addressed in this manuscript. Today, more than 6 000 GMNs of microorganisms and multicellular organisms, such as humans or plants, have been reconstructed (Passi et al., 2021). Given the availability of high-quality metabolic networks in public databases, such as BiGG (King et al., 2015), we consider metabolic networks to be inputs for our methods. We do not need to infer metabolic networks by ourselves to build regulated metabolic networks.

2.2 Inference of Boolean Networks

Many methods have been developed to infer Boolean networks (BN) from *experimental observations* and *prior knowledge networks* (PKN), sets of signed interactions between genes, proteins, and metabolites. The methods developed so far only rely on the information on the regulatory scale of the cell, mainly transcriptomics, proteomics, and phosphoproteomics. They typically employ combinatorial (Saez-Rodriguez et al., 2009; Ostrowski et al., 2016; Videla et al., 2017; Razzaq et al., 2018; Chevalier et al., 2020; Vaginay et al., 2021) or continuous (Terfve et al., 2012; Tsiantis et al., 2018) formulations to optimize the data-fitting and parsimony hypotheses.

Methods. The main differences between these methods lie in their input data and how they define the compatibility between the dynamics of a BN and the input observations. Tools like `CASPO` (Videla et al., 2017) and `CellNOptR` (Terfve et al., 2012) infer BNs from steady-state observations of gene expressions, and define the compatibility as the observation being a fixpoint of the BN under synchronous update semantics. Other methods, such as `CaspoTS` (Ostrowski et al.,

2016), `BoNesis` (Chevalier et al., 2020), and ASKEed (Vaginay et al., 2021), infer BNs compatible with multiple time series observations by defining compatibility through a reachability condition: for each time series, there must exist a sequence of configurations (for a chosen update semantics) that match the observations. Except for `CellNOptR`, all these methods rely on *Answer Set Programming* (ASP) (Baral, 2003; Gebser et al., 2012), a logic programming framework for expressing symbolic satisfiability problems[2], to infer BNs that best fit the input observations and are supported by the PKN. These ASP-based tools can enumerate all the solution regulatory rules, providing a comprehensive view of the solution spaces, which is essential for biologists who need insights into the space of feasible regulatory rules to enhance their understanding of biological systems and plan their experiments.

Limits. The main drawback of methods like `CaspoTS` and `BoNesis` is that they rely on a simplified assumption about the regulatory system dynamics: interactions with the metabolism are totally neglected. For example, if we apply the inference procedure of `CaspoTS` on simulated data generated from rFBA simulation of the core-carbon metabolism (Covert et al., 2001), multiple equivalent BNs are inferred, but only one accurately reproduces the expected behaviors.

Stochastic methods. Note that stochastic inference methods, based on genetic algorithms (Trinh and Kwon, 2021; Gao et al., 2020; Liu et al., 2021) or deep learning (Barman and Kwon, 2020), are also used to infer BNs. However, these methods do not exploit prior knowledge of regulatory interactions to infer BNs. The inferred BNs may not be limited to manually curated or high-confidence interactions. Furthermore, these methods cannot characterize the set of valid regulatory rules, as they are intrinsically unable to infer the complete set of solutions. For these reasons, we will not consider stochastic inference methods in this work.

2.3 Inference of Boolean Networks Controlling Metabolic Networks

In the previous sections, we reviewed methods for reconstructing metabolic networks and inferring BNs of regulatory systems. To reconstruct regulated metabolic networks (RMN), the current bottleneck lies in the inferring of the metabolic feedback and control regulatory rules. Given the absence of methods for inferring these rules, we can only outline the problem's definition based on the methods used for reconstructing metabolic networks and inferring BNs.

Problem definition. The problem of inferring metabolic feedback and control rules comes down to inferring BNs that control a metabolic network. This problem

[2] ASP is detailed in Section 3.1.

would take as input: **(i)** a metabolic network; **(ii)** a PKN, that is a set of *a priori* interactions between genes, proteins, and the metabolism; and **(iii)** multiple time series of omics data. Since there are efficient protocols for reconstructing metabolic networks and databases that contain them, they can be considered as inputs.

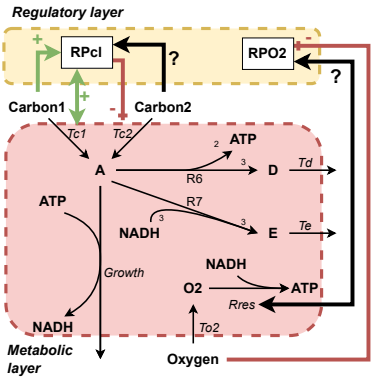
The objective of the inference problem would be to automatically infer BNs controlling the input metabolic network such that:

1. the BNs are supported by the PKN;
2. the coupled dynamic of the BN and the metabolic network is compatible with the input omics data.

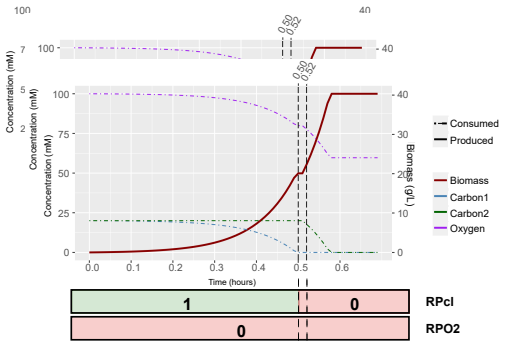
The first constraint is common in BN inferring approaches, it ensures that inferred BNs are biologically relevant. Unlike traditional BN inference approaches, this inference problem would consider the coupled dynamics of BNs and metabolic networks, with formalism such as the regulatory flux balance analysis (rFBA), to ensure compatibility with the input omics data.

Type of omics data. To infer BNs that control metabolic networks, observations of both the regulatory and metabolic scales are necessary. In this manuscript, we utilize time series of *kinetics*, *fluxomics*, and *transcriptomics* data. Time series of transcriptomics data provide information about the dynamics of the regulatory system and are commonly used to infer Boolean regulatory rules. Time series of kinetics and fluxomics data provide information about the dynamics of the metabolism. In particular, kinetics and fluxomics are used to build and calibrate metabolic networks by ensuring the compatibility of the observations with the FBA (Thiele and Palsson, 2010). Consequently, the reconstructed input metabolic network will be compatible with these data types. In practice, we consider that the observed time series contain at least two observations per growth phases.

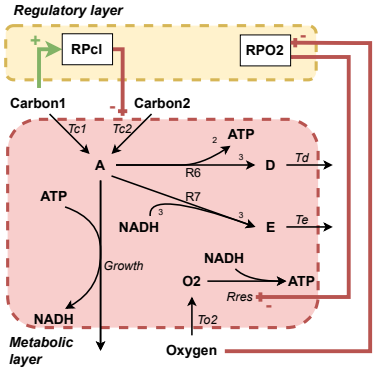
Example. An example of inference problem instance is shown in Fig. 12. The input metabolic network, as described in Fig. 3, along with eight *a priori* interactions are shown in Fig. 12a, and the time series data in Fig. 12b. Among the eight *a priori* interactions, there are three positive interactions ((Carbon1, +, RPcl), (Tc1, +, RPcl), and (RPcl, +, Tc1)), two negative interactions ((RPcl, -, Tc2) and (Oxygen, -, RPO2)), and three unsigned interactions, *i.e.* interactions that can either be positive or negative ((Carbon2, ?, RPcl), (Rres, ?, RPO2), and (RPO2, ?, Rres) where $? \in \{-, +\}$). Two examples of Boolean networks that control the input metabolic networks are presented in Fig. 12c, non-constant Boolean rules are provided under the figures. Both Boolean networks are consistent with the input interactions, although not all interactions are utilized (*e.g.* (Carbon2, ?, RPcl)), and the associated RMNs admit rFBA simulations compatible with the input time series data.



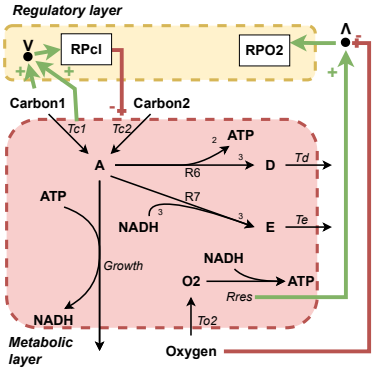
(a) Input metabolic network $\mathcal{N}$ from Fig. 3, and a set of 8 prior knowledge interactions (green arrows: positive; red arrows: negative; bold black arrows: undefined).



(b) Multiple input time series of observations of the activity of the regulatory and metabolic layers.



$f_{\mathrm{RPcl}}(x) = x_{\mathrm{Carbon1}}$ $f_{\mathrm{Tc2}}(x) = \neg x_{\mathrm{RPcl}}$
$f_{\mathrm{RPO2}}(x) = \neg x_{\mathrm{Oxygen}}$ $f_{\mathrm{Rres}}(x) = \neg x_{\mathrm{RPO2}}$



$f_{\mathrm{RPcl}}(x) = x_{\mathrm{Carbon1}} \vee x_{\mathrm{Tc1}}$ $f_{\mathrm{Tc2}}(x) = \neg x_{\mathrm{RPcl}}$
$f_{\mathrm{RPO2}}(x) = \neg x_{\mathrm{Oxygen}} \wedge x_{\mathrm{Rres}}$ $f_{\mathrm{Rres}}(x) = 1$

(c) Example of 2 solution BNs: they are supported by the prior interactions and fit with input time series. The left regulated metabolic network (RMN) is shown in Fig. 8; the right RMN is composed of another feasible set of regulatory rules explaining a diauxic shift (Fig. 9a).

■ **Figure 12** – Example of an instance of the Boolean network inference problem from multiple time series of fluxomics, kinetics, and/or transcriptomics observations. The **inputs** are composed of (a) a metabolic network $\mathcal{N}$ and a set of prior knowledge interactions; (b) multiple time series fluxomics, kinetics, and/or transcriptomics observations. Green (*resp.* red) arrows are positive (*resp.* negative) interactions. Dark arrows with question marks are unsigned interactions, *i.e.* either positive or negative. **Solutions to this inference problem** are Boolean networks f supported by the prior knowledge interactions for which there exist rFBA traces of the regulated metabolic network $(\mathcal{N}, \mathcal{P}, f)$ compatible with each input observation (c).

3 Solving Hybrid Satisfiability and Optimization Problems

In this manuscript, we formulate the inference of Boolean networks (BN) controlling a metabolic network as a hybrid optimization problem that merges combinatorial and quantified linear constraints. In practice, combinatorial problems are solved with SAT solvers or ASP solvers. ASP has been widely used in systems biology to address highly combinatorial problems. In particular, most of the BN inference methods described in the previous section rely on ASP (Videla et al., 2017; Ostrowski et al., 2016; Chevalier et al., 2020; Vaginay et al., 2021). Moreover, ASP and its linear extensions have been applied to solve problems related to the metabolism dynamics, where the FBA must be satisfied. Examples include metabolic network gap-filling (Prigent et al., 2017; Frioux et al., 2019), or elementary flux modes enumeration (Mahout et al., 2020).

In this section, we first present ASP (Section 3.1). Then, we present its extensions used to address linear arithmetic constraints (Section 3.2).

3.1 Answer Set Programming (ASP)

ASP is a declarative logic framework that allows for solving combinatorial satisfiability and optimization problems (Baral, 2003; Gebser et al., 2012). The core idea of ASP is to model a search problem in a logical format, *i.e.* as a set of logical rules so that the models of the logic problem represent the solutions to the original problem. Stable models of the logic programs are referred to as *answer sets*. Modern ASP solvers, like *clingo* (Gebser et al., 2017), can solve and enumerate the answer sets of NP problems with millions of variables. Although determining whether a program has an answer set is the fundamental decision problem in ASP, ASP solvers support various combinations of reasoning modes, among them, regular and projective enumeration, intersection and union, multi-criterion optimization, and subset minimal model enumeration (Gebser et al., 2011, 2013).

In this section, we introduce the basics of the ASP syntax (Section 3.1.1). Then, we explain the stable model semantics used to solve ASP programs (Section 3.1.2).

3.1.1 ASP's Syntax

Rule syntax. An ASP program is a set of logical rules of the form:

$$a_0 \text{ :- } a_1, \cdots, a_i, \text{ not } a_{i+1}, \cdots, \text{ not } a_n.$$

where $\{a_0, \cdots, a_n\}$ are atoms. In practice, the left-hand part of the rule (a_0), preceding ':-', is called the *head*, and the right-hand part ($a_1, \cdots, a_i$, not $a_{i+1}, \cdots$, not a_n) is called the '*body*'. A rule can be intuitively understood as "if the body of

the rule holds ($\{a_1, \cdots, a_i\}$ are true and $\{a_{i+1}, \cdots, a_n\}$ are false), then the head of the rule (a_0) is true". In ASP, an atom can be true only if it appears in the head of a rule whose body is satisfied.

Facts and integrity constraints. From this general definition of logical rules, we distinguish two special types of rules: facts and integrity constraints. A *fact* is a logical rule with an empty body:

$$a_0 \text{ :- } .$$

A fact models an input knowledge and asserts that its head (a_0) is always true. An *integrity constraint* is a logical rule whose head is empty, or a_0 is $\bot$ (*false*):

$$\text{:- } a_1, \cdots, a_i, \text{ not } a_{i+1}, \cdots, \text{ not } a_n.$$

An integrity constraint ensures that any model satisfying the rule's body is excluded, filtering out models that do not meet certain conditions.

Grounding. In practice, ASP programs are rewritten using templates similar to first-order logic predicates. Within rules, atoms are predicate symbols followed by a sequence of terms (*e.g.* $f(c)$, $p(f(X), Y, 0)$) and terms are constants (*e.g.* c, 0), function symbols followed by terms (*e.g.* $f(X)$), or variables (*e.g.* X, Y).

Before solving, the program atoms are instantiated on the universe of possibilities. This instantiation phase is called the '*grounding*' of the ASP program, and can generate an exponential number of variable-free logical rules.

Consider the two facts v(1). and v(2)., and the rule $p(X, Y)$:- $\text{v}(X), \text{v}(Y), X! = 1$. where X and Y are variables. This rule is *grounded* into four variable-free logical rules, representing all combinations of values for X and Y according to the universe of possibilities ($v(1)$ and $v(2)$):

$$p(1,1) \text{ :- } v(1), v(1), 1 \text{ != } 1. \qquad p(1,2) \text{ :- } v(1), v(2), 1 \text{ != } 1.$$
$$p(2,1) \text{ :- } v(2), v(2), 2 \text{ != } 1. \qquad p(2,2) \text{ :- } v(2), v(2), 2 \text{ != } 1.$$

Modern grounders aim to reduce the number of logical rules generated during grounding by not generating trivial rules (Kaufmann et al., 2016), that is, rules for which the body could never be satisfied. In the previous example, the first two grounded rules are not generated in practice, since their body cannot be satisfied ($1 \text{ != } 1$ is necessarily false).

3.1.2 Stable Model Semantics

Models of ASP programs are sets of grounded atoms, called *answer sets*. Answer sets are stable models that adhere to the *stable model semantics* (Gelfond and Lifschitz,

1988). This semantics assumes that all atoms are false by default. Consequently, each atom that appears in the answer set is considered true, while all atoms not included in the answer set are considered false.

The stable model semantics requires each atom in an answer set to be derived from a fact or the head of a rule whose body is satisfied by the answer set, and the answer set to be subset minimal. In other words, a stable model M for an ASP program P is such that M is a subset minimal set of grounded atoms satisfying the reduced ASP program P^M, where P^M is obtained by removing rules with negated atoms not in M and removing the negated atoms from the remaining rules. Determining if an ASP program has at least one answer set is NP-complete.

Example. Let us consider the following ASP program P:

$$a \text{ :- } . \tag{I.4}$$
$$b \text{ :- not } c, a. \tag{I.5}$$
$$c \text{ :- not } d. \tag{I.6}$$

Suppose that we want to check if $M_1 = \{a, b\}$ is an answer set of P. First, we compute the reduced program $P^{\{a,b\}}$ by dropping the negated atoms (not c) in the second rule (Eq. I.5) and removing the last rule (Eq. I.6):

$$a \text{ :- } .$$
$$b \text{ :- } a.$$

We can see that M_1 is the only stable model of $P^{\{a,b\}}$, therefore it is also a stable model of P. In the same way, we can show that $M_2 = \{a, b, c\}$ is not a stable model of P. The reduced program $P^{\{a,b,c\}}$ is only composed of the rule a :- .. Therefore, the smallest answer set of $P^{\{a,b,c\}}$ is $M' = \{a\}$ which is a subset of M_2.
The ASP program P has two answer sets: $\{a, b\}$ and $\{a, c\}$.

3.2 ASP Modulo Theory Extensions

Extensions of ASP have been proposed to extend ASP solvers with constraints from other theories. One notable extension is *clingo[LP]* (Janhunen et al., 2017), which extends the standard ASP solver *clingo* (Gebser et al., 2017) with quantifier-free linear constraints over real-valued variables.

Clingo[LP]. This extended solver has been used to solve hybrid combinatorial problems in systems biology, including problems that necessitate the FBA equations to be satisfied (Frioux et al., 2019; Mahout et al., 2020). *Clingo[LP]* extends *clingo* by incorporating constraint propagation to handle linear constraints alongside the

logical rules of ASP. During the ASP-solving process, a linear optimization problem is dynamically constructed based on the grounded atoms within the current answer set. At different stages of the solving process, a dedicated linear solver (such as CPLEX or LpSolve for *clingo[LP]*) is called to check the satisfiability of the linear constraints. If the current set of linear constraints is found to be unsatisfiable, a *nogood* (a constraint similar to an integrity constraint) is added to the ASP program, rejecting the current (partial) answer set. When a (partial) answer set is rejected, the ASP solver backtracks to a previously valid state. This process is repeated until either the hybrid ASP program is proven unsatisfiable or an answer set, whose associated linear problem in the linear propagator is satisfiable, is found. To facilitate the integration of linear constraints in ASP programs, *clingo[LP]* introduces an extended ASP syntax.

Recently, a novel extension named *clingo-lpx* has been introduced. This extension is available on GitHub[3]. Unlike *clingo[LP]*, *clingo-lpx* relies on a dedicated implementation of the simplex algorithm (Dutertre and De Moura, 2006) to ensure the satisfiability of linear constraints. In practice, *clingo-lpx* outperforms *clingo[LP]* regarding computation times. In Chapter V, we compare our hybrid ASP solving framework against *clingo-lpx*.

3.2.1 Custom Theory Propagators with *clingo*

In practice, *clingo* provides a standard interface to integrate custom theory propagators into its ASP solving process (Ostrowski and Schaub, 2012; Banbara et al., 2017). Here, we briefly describe the main functions to implement such a theory propagator using *clingo* Python API. A comprehensive guide for building custom ASP-based systems is available in Kaminksi et al. (2023).

A theory propagator must implement four functions to be integrated in *clingo*: `initialize`, `propagate`, `undo`, and `check`. These functions are invoked by *clingo* at different stages of the solving process to communicate the current assignment of atoms (partial answer sets) to the theory propagator. The ASP solving workflow with theory propagator is described in Fig. 13.

`initialize`: This function is called once at the beginning of the solving process. It is used to initialize the theory propagator and to declare atoms of interest, referred to as *watched atoms*.

`propagate`: This function is called each time the ASP solver decides the state (true or false) of a watched atom. It is used to track the state of the partial answer set and to check its satisfiability with respect to the theory constraints. For *clingo[LP]*, linear constraints associated with theory atoms decided true are

[3]Available on git: https://github.com/potassco/clingo-lpx.

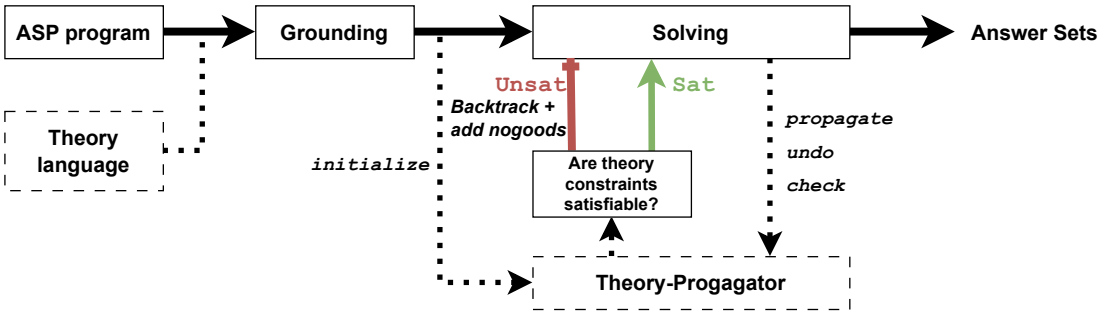


Figure 13 – Workflow of the ASP solving process with and without theory propagation. Solid-lined boxes and arrows represent the standard ASP-solving process. Dotted-lined boxes and arrows, and colored arrows, represent the processes related to theory propagation. The functions `initialize`, `propagate`, `undo`, and `check` are part of the *clingo* API and are called during the solving process to manage theory propagation. The complete solving process is as follows: first, the ASP program and the theory language are grounded. Then, the theory propagator is initialized, and the solving of the grounded ASP program begins. The theory propagator is invoked through the functions `propagate`, `undo`, and `check` whenever the state of an atom related to the theory propagator is updated. The theory constraints associated with the current answer set are checked. If they are unsatisfiable, the solving process backtracks, and a nogood is generated. Otherwise, the solving process continues, until either the grounded ASP program is found unsatisfiable regarding the theory constraints or answer sets satisfying the theory constraints are found.

added to the linear optimization problem. This linear problem is then solved to check the satisfiability of the current set of linear constraints. If they are not satisfiable, nogoods are generated and added to the ASP solver.

`undo`: This function is called each time that the ASP solver backtracks the state of a watched atom. For *clingo[LP]*, the linear constraints associated with the backtracked theory atoms are removed from the linear optimization problem.

`check`: This function is only called when all atoms' states have been decided, that is when the ASP solver has found an answer set of the ASP program that satisfies all the nogoods generated so far. Watched atoms can have been decided between the last call to `propagate` and the call to `check`. Therefore, it is necessary to ensure that the theory constraints are still satisfied, even if all checks made in previous calls to the `propagate` function passed.

Depending on the application, it is not always necessary to define a theory language extension for the ASP syntax. Such extensions are beneficial when developing a generic ASP modulo theory solver, as they ensure the syntax of theory constraints. However, they may be unnecessary when developing dedicated solving

methods where users will not interact with the hybrid ASP program directly. For instance, in Chapter IV, we introduce a dedicated implementation for solving the inference problem based on a hybrid extension of ASP. We do not extend the ASP syntax for it. Conversely, in Chapter V, we introduce a generic ASP modulo quantifier linear arithmetic solver for which we propose a syntax extension of ASP, based on the one introduced in *clingo[LP]*.

3.2.2 Satisfiability Modulo Theory Solvers

In practice, hybrid satisfiability or optimization problems that merge logical constraints and linear constraints are *Satisfiability Modulo Theory* (SMT) problems (Barrett and Tinelli, 2018). In particular, problems that merge logic and linear constraints are satisfiability modulo linear real arithmetic (LRA) problems. There exist different solvers for solving SMT problems, one of the most well-known SMT solvers being *z3* (De Moura and Bjørner, 2008). These SMT solvers extend SAT solvers with theory-dependent constraints propagation methods. For the LRA theory, the SMT solvers rely on a dedicated implementation of the simplex algorithm to check the satisfiability of linear constraints (Dutertre and De Moura, 2006). Unlike the theory propagator used to extend ASP, SMT solvers with the LRA theory generate new linear constraints (while ASP's theory propagator generates new logic/integrity constraints) (Farzan and Kincaid, 2016; Reynolds et al., 2017).

In this manuscript, we do not rely on SMT solvers to model and solve the inference problem, but on hybrid ASP solvers. The choice of ASP is motivated by its known performance for enumerating the solutions of highly combinatorial problems, even compared to well-established SMT solvers like *z3* (Gebser et al., 2014). In addition, most SMT solvers do not handle optimization constraints and quantifiers, which is necessary for solving our formulation of the Boolean network inference problem.

4 Thesis Contributions

In this manuscript, we tackle the problem of inferring Boolean regulatory rules controlling metabolic networks from time series kinetics, fluxomics, and transcriptomics observations.

Thesis organization. The manuscript is organized into seven chapters, including the introduction, the state-of-the-art, and the conclusion. First, in Chapter II, we formally define the problem of inferring Boolean regulatory rules controlling a metabolic network from time series observations. We introduce a general definition of this problem as a hybrid optimization problem, specifically a combinatorial optimization problem with quantified linear constraints. From this general definition, we derive three formulations, each corresponding to different levels of abstraction of the rFBA dynamics. The three following chapters are each based on a published paper, in which we introduce the problem abstraction as well as ASP-based solving methods to address them.

In Chapter III and the associated paper Thuillier et al. (2021), we introduce a Boolean abstraction of the rFBA dynamics. This abstraction allows us to relax the inference problem as a pure Boolean combinatorial satisfiability problem with two levels of quantifiers, known as 2-QBF problems. We solve it using an ASP-based implementation that leverages the stable model semantics and disjunctive logic extension of ASP[4] to efficiently enumerate solutions. The Boolean abstraction of rFBA introduced in this chapter is the foundation of all the solving methods developed throughout this thesis.

In Chapter IV and the associated paper Thuillier et al. (2022), we formulate and solve the inference problem as a hybrid optimization problem merging logical constraints and linear constraints, specifically the FBA equations. We develop a dedicated ASP-based solving method, integrating the FBA equations and growth maximization within the ASP solving process, to address it. We highlight the efficiency of our inferring method on the model of core-carbon metabolism (Covert et al., 2001) for which we generate a benchmark of realistic *in silico* fluxomics, kinetics, and transcriptomics time series.

In Chapter V and the associated paper Thuillier et al. (2024), we extend our solving workflow to a broader class of hybrid problems, namely hybrid combinatorial optimization problems under logic and quantified linear constraints (OPT+qLP). The hybrid formulation of the inference problem is an example of OPT+qLP problem. Here, we present a generic and efficient method to address OPT+qLP problems, providing a versatile solution applicable beyond the specific context of Boolean regulatory rules inference.

[4]Described in Chapter III.

Appendices. The contributions of this manuscript are completed with additional data and results. In Appendix A, we provide a detailed description of two regulated metabolic networks of *Escherichia coli*: a model of core-carbon metabolism (Covert et al., 2001) (used in Chapter III to V); and a medium-scale model (Covert and Palsson, 2002) (used in Chapters IV and V). These networks were only described in their introductory paper appendices and lacked standard file representations. A significant portion of the thesis has been dedicated to formatting these networks into standard file formats suitable for rFBA simulation tools, namely *SBML* (Hucka et al., 2003) for metabolic networks and *SBML-qual* (Chaouiya et al., 2013) for Boolean networks.

In Appendix B, we describe the ASP, and hybrid ASP, encodings of the relaxed inference problem defined in Chapter III and the hybrid inference problems defined in Chapters IV and V.

In Appendix C, we present a second case-study application of the relaxed inference problem (Chapter III) applied to the model of core-carbon metabolism.

II Formalization of the Inference of Boolean Networks Controlling Metabolic Networks From Time Series Data

To sum up

In this chapter, we introduce a general definition of the inference of Boolean networks controlling metabolic networks from time series data. In particular, we consider observations from both the regulatory and metabolic scales: transcriptomics, and kinetics and fluxomics, respectively. From this general definition, we derive three formulations of the inference problem based on different levels of abstraction of the rFBA formalism.

In this chapter

In this chapter, we introduce a novel definition of the inference problem from time series observations of both regulatory and metabolic scales. Our formulation of the problem takes as input **(i)** a metabolic network; **(ii)** a prior knowledge network (PKN), *i.e.* a domain of putative influences; and **(iii)** a set of time series observations, namely kinetics, fluxomics, and transcriptomics. By modeling the

metabolism dynamics directly in the inference problem, we aim to infer Boolean regulatory rules, including the feedback and control rules, controlling the metabolism. The output of the problem is a set of Boolean networks (BN) that are supported by the PKN and that best explain the input time series.

Outlines. First, we formalize the problem input: the PKN in Section 1.1 and time series data in Section 1.2. Then, we define the compatibility between time series data and a regulated metabolic network (RMN) in Section 2. Finally, we propose a general definition of the inference problem as a combinatorial optimization problem under quantified constraints in Section 3. In this last section, we introduce three formulations of the inference problem. These formulations are explained in further detail in the next chapters.

1 Input of the Inference Problem

1.1 Prior Knowledge Network

Influences. Let u and v be two components implied in the regulations. The influence of u on v is denoted by the triplet (u, s, v) where $s \in \{+; -\}$ is the sign of the influence. The sign is $s =$ '+' if u is an activator of v, else it is $s =$ '$-$' with u an inhibitor of v. The set of all influences defining the domain of putative influences of the inference problem is called a *prior knowledge network*.

▶ **Definition 1.1: Prior Knowledge Network (PKN)**

> A *prior knowledge network* (PKN) is an influence graph $\mathcal{G}$ constraining the search domain of regulatory networks.

For the rest, we assume that there is no influence on external metabolites, *i.e.* $\forall (u, s, v) \in \mathcal{G}, v \notin \mathcal{M}_{\text{ext}}$. Indeed, the state of an external metabolite $m \in \mathcal{M}_{\text{ext}}$ is only dependent on its availability in the cell environment.

Search domain. To be biologically relevant, the inferred BNs should be supported by the PKN, that is, regulatory rules should only rely on the input interactions to explain gene expressions. A BN f is said to be *supported* by a PKN $\mathcal{G}$ if its influence graph $G(f)$ is a subgraph of $\mathcal{G}$, *i.e.* $G(f) \subseteq \mathcal{G}$.

Figure 14 – Example of a prior knowledge network (PKN) of dimension 3. It is composed of 3 regulatory components: M, R, and RP; and 3 influences: $(M, +, RP)$, $(R, -, RP)$, and $(RP, +, R)$. The green arrows are positive influences, and the red arrow is a negative influence.

▶ **Definition 1.2: Search domain for regulatory networks $\mathbb{F}(\mathcal{G})$**

> Let $\mathcal{G}$ be a PKN of dimension n. The *search space* $\mathbb{F}(\mathcal{G})$ contains all Boolean networks f of dimension n whose influence graph $G(f)$ is a subgraph of the PKN $\mathcal{G}$, *i.e.* $\forall f \in \mathbb{F}, G(f) \subseteq \mathcal{G}$.
> The size of $\mathbb{F}(\mathcal{G})$ is doubly exponential in n.

Example. Let consider the PKN $\mathcal{G} = (\{M, R, RP\}, \{(M, +, RP), (R, -, RP), (RP, +, R)\})$ of dimension $n = 3$ described in Fig. 14. In this example, there are three regulatory components (M, R, and RP a metabolite, a reaction, and a regulatory protein, respectively) and three influences ($(M, +, RP)$, $(R, -, RP)$, and $(RP, +, R)$). For each component i, the constant Boolean rules $f_i(x) = 1$ (always true) and $f_i(x) = 0$ (always false) are compatible with any influence.

There is no influence toward M, the only rules compatible with $\mathcal{G}$ are the 2 constant rules: $f_M(x) = 1$ and $f_M(x) = 0$. The regulatory state of the reaction R is positively influenced by the state of the regulatory protein RP ($(RP, +, R)$). There are 3 regulatory rules compatible with this influence, the two constant rules $f_R(x) = 1$ and $f_R(x) = 0$, and $f_R(x) = x_{RP}$. Finally, the regulatory state of the regulatory protein RP is positively influenced by the availability of M in the bacteria substrate ($(M, +, RP)$), and negatively by the activity of the reaction R ($(R, -, RP)$). There are 6 regulatory rules compatible with these two influences: the two constant rules $f_{RP}(x) = 1$ and $f_{RP}(x) = 0$, and any Boolean rules that are a combination of x_M (positive influence of M) and $\neg x_R$ (negative influence of R) with the logical 'and' ($\wedge$) and logical 'or' ($\vee$) operators ($f_{RP}(x) = x_M$, $f_{RP}(x) = \neg x_R$, $f_{RP}(x) = x_M \vee \neg x_R$, and $f_{RP}(x) = x_M \wedge \neg x_R$).

The search space $\mathbb{F}(\mathcal{G})$ contains all regulatory networks for which all regulatory rules are compatible with $\mathcal{G}$. There are therefore $2 \times 3 \times 6 = 36$ regulatory rules compatible with the 3 influences of the PKN $\mathcal{G}$.

1.2 Observations

In this work, we consider time series observations of kinetics, fluxomics, and/or transcriptomics data.

Transcriptomics data. Transcriptomics data are qualitative observations of the regulatory scale. They are measures of the expression of gene activities, specifically, they quantify the mRNA produced during gene transcriptions. From transcriptomics data, information about the gene expressions, the regulatory proteins' availabilities, and reactions' activity states (on/off) can be inferred.

In this manuscript, we assume that the expression data are qualitative observations of the reactions ($\mathcal{R}$) states, and of the genes and regulatory proteins ($\mathcal{P}$) states. Moreover, we further assume that the availability states of external metabolites ($\mathcal{M}_{\text{ext}}$) is also measured. Transcriptomics data can be, therefore, modeled as a Boolean vector $\hat{x} \in \mathbb{B}^{|\mathcal{P}|+|\mathcal{M}_{\text{ext}}|+|\mathcal{R}|}$.

Kinetics and fluxomics. Kinetics and fluxomics are two quantitative observations of the metabolic scale. Kinetics measures the concentrations of external metabolites ($\mathcal{M}_{\text{ext}}$). Fluxomics quantifies the activity rates of reactions ($\mathcal{R}$). Specifically, it quantifies the reactions' conversion rates, *i.e.* the amount of reactant metabolites converted into product metabolites in a given amount of time. For the rest, we denote by $\hat{w} \in \mathbb{R}^{|\mathcal{M}_{\text{ext}}|}$ and $\hat{v} \in \mathbb{R}^{|\mathcal{R}|}$ the kinetics and fluxomics observations, respectively.

Observation. An observation can be composed of any combination of transcriptomics, kinetics, and fluxomics data. All unobserved elements are set to an undefined value '$\perp$'. For instance, an observation composed of kinetics ($\hat{w}$) and transcriptomics ($\hat{x}$) does not have any information about reactions' activities ($\hat{v}$). The fluxomics observations are set to undefined, *i.e.* $\hat{v} = \{\perp\}^{|\mathcal{R}|}$.

Let an observation be a triplet $o = (\hat{v}, \hat{w}, \hat{x})$ where $\hat{v}$ are fluxomics data, $\hat{w}$ are kinetics data, and $\hat{x}$ are transcriptomics data. Following this definition, an observation can be seen as a partial regulated metabolic steady-state[1] (RMSS). Given an RMSS (v, w, x), fluxomics data $\hat{v}$ provide information about the metabolic state v; kinetics data $\hat{w}$ provide information about the substrate state w; and transcriptomics data $\hat{x}$ provide information about the regulatory state x.

Let $\mathbb{R}_\perp$ (*resp.* $\mathbb{B}_\perp$) be the set of real (*resp.* Boolean) values or the undefined value '$\perp$'. For the rest, we formally define an observation as a partial RMSS for which unobserved components are set to the undefined value ('$\perp$').

[1] See Definition 1.6 in Chapter I

Data types	scale	Modality	Observed components $\mathcal{R}$	$\mathcal{M}_{\text{ext}}$	$\mathcal{P}$	Observation $(\hat{v},$	$\hat{w},$	$\hat{x})$
Kinetics	Metabolic	Quantitative	-	✓	-	-	✓	-
Fluxomics	Metabolic	Quantitative	✓	-	-	✓	-	-
Transcriptomics	Regulatory	Qualitative	✓	✓	✓	-	-	✓

Table 4 – Summary of the structure of observations for the considered data types: kinetics, fluxomics, and transcriptomics. The modality is the nature of the observations: quantitative for real-valued observations and qualitative for Boolean-valued observations. For observed components, there is a checkmark ('✓') if the set of elements is observed, while there is a dash ('−') if it is not observed. For observations, a check mark ('✓') represents defined elements, while a dash ('−') represents undefined ones.

▶ **Definition 1.3: Observation**

An *observation* is a triplet $o = (\hat{v}, \hat{w}, \hat{x}) \in \mathbb{R}_{\perp}^{|\mathcal{R}|} \times \mathbb{R}_{\perp}^{|\mathcal{M}_{\text{ext}}|} \times \mathbb{B}_{\perp}^{|\mathcal{P}|+|\mathcal{M}_{\text{ext}}|+|\mathcal{R}|}$ representing a partial metabolic steady-state.

The information provided by each considered omics data is summarized in Table. 4.

Time series observation. Time series observations are a sequence of successive observations of a biological system obtained during an experiment.

▶ **Definition 1.4: Observed time series**

An *observed time series* $\mathcal{T}_o = \{o^1, \cdots, o^m\}$ is a sequence of $m \geq 1$ successive observations.

2 Compatiblity of RMN Traces with Observations

In this section, we define the compatibility of an RMN $(\mathcal{N}, \mathcal{P}, f)$ with an observed time series $\mathcal{T}_o = \{(v_o^t, w_o^t, x_o^t)\}_{t=0}^m$. We assume that the RMN dynamics is modeled by successive RMSSs of the form (v, w, x) with $v \in \mathbb{R}^{|\mathcal{R}|}$ the metabolic state, $w \in \mathbb{R}^{|\mathcal{M}_{\text{ext}}|}$ the concentration of external metabolites, and $x \in \mathbb{B}^{|\mathcal{P}|+|\mathcal{M}_{\text{ext}}|+|\mathcal{R}|}$ the regulatory state.

Outlines. First, in Section 2.1, we define the compatibility between an observation and a RMSS. Then, in Section 2.2, we define the compatibility of an RMSS with the metabolic growth phenotype described by fluxomics observations. Finally, in

Section 2.3, we propose a general definition for the compatibility of an RMN trace with an observed time series. This definition is agnostic of the simulation framework chosen to model the RMN dynamics.

2.1 Compatibility Between an Observation and an RMSS

Regulated metabolic state consistency. A regulated metabolic steady-state (RMSS) (v, w, x) is said *consistent* if the regulatory states of reactions and external metabolites are compatible with the metabolic and substrate states. For a reaction $r \in \mathcal{R}$, this implies that the regulatory state x_r should not inhibit the reaction if it has a non-null metabolic flux, *i.e.* if $v_r > 0$ (Eq. II.1a). There is no constraint on x_r if v_r is null. For an external metabolite $m \in \mathcal{M}_{\text{ext}}$, it implies that the metabolic state x_m corresponds to the availability of m in the cell environment (Eq. II.1b).

Let $\beta : \mathbb{R} \to \mathbb{B}$ be a binarization function such that $\forall\, s \in \mathbb{R}$, $\beta(s) = 1$ if and only if $s_i \neq 0$, else $\beta(s)_i = 0$.

▶ **Definition 2.1: Consistent RMSS**

An RMSS (v, w, x) is said *consistent* if and only if its regulatory state x is compatible with its metabolic state v and substrate state w, *i.e.* it satisfies Eqs. II.1.

$$\forall\, r \in \mathcal{R},\ x_r \geq \beta(v_r) \tag{II.1a}$$

$$\forall\, m \in \mathcal{M}_{\text{ext}},\ x_m = \beta(w_m) \tag{II.1b}$$

Compatibility with observations. To ensure the compatibility between an observation $(\hat{v}, \hat{w}, \hat{x})$ and an RMSS (v, w, x), it is essential to consider experimental errors. We assume that all the low-confidence observations are removed and replaced by an undefined value ('$\perp$').

For the RMSS to be compatible with the observation, the kinetics and transcriptomics must match exactly with the substrate and regulatory state, respectively. Indeed, we assume that regulatory rules influenced by external metabolites depend solely on the presence of these metabolites rather than specific concentration thresholds. Therefore, noise in the kinetics data will not affect the inference of regulatory rules, assuming that the availability states of external metabolites are valid. Moreover, we assumed that transcriptomics data contain only high-confidence observations, therefore, the inferred Boolean networks should exactly reproduce these observations.

Regarding fluxomics observations, we only consider the observed reaction activity states to mitigate the impact of experimental noise. If a reaction is observed active

(*resp.* inactive), then the reaction should have a non-zero flux (*resp.* a null flux) in the metabolic state, *i.e.* $\forall r \in \mathcal{R},\ (\hat{v}_r > 0 \implies v_r > 0) \wedge (\hat{v}_r = 0 \implies v_r = 0)$. As for kinetics, we assume that feedback rules depend solely on the activity states of these reactions rather than specific thresholds on metabolic activity. Hence, this definition ensures that the metabolic state is compatible with the observed metabolic phenotype described by fluxomics data, reducing the impact of experimental error in the observations.

▶ **Definition 2.2: Data compatibility**

An observation $(\hat{v}, \hat{w}, \hat{x})$ is data-compatible with a consistent RMSS (v, w, x) if and only if **(i)** kinetics $\hat{w}$ and expression data $\hat{x}$ match exactly with the substrate state w and regulatory state x, respectively (Eqs. II.2a and II.2b), and **(ii)** the supports of the metabolic state and the fluxomics observations are the same (Eq. II.2c).

$$\forall m \in \mathcal{M}_{\text{ext}},\ \hat{w}_m \neq \bot \implies w_m = \hat{w}_m \tag{II.2a}$$
$$\forall p \in \mathcal{P} \cup \mathcal{M}_{\text{ext}} \cup \mathcal{R},\ \hat{x}_p \neq \bot \implies x_p = \hat{x}_p \tag{II.2b}$$
$$\forall r \in \mathcal{R},\ \hat{v}_r \neq \bot \implies (v_r > 0 \iff \hat{v}_r > 0) \tag{II.2c}$$

2.2 Compatibility with an Observed Growth Phenotype

Determining the regulatory states of inactivated reactions, *i.e.* reactions without metabolic activity, is not straightforward. A reaction can be inactivated by an inhibition from control rules or due to metabolic-related constraints. It is necessary to retrieve the exact regulatory states of inactivated reactions to accurately infer regulatory rules.

Phenotype compatibility. The exact regulatory states of inactivated reactions can be deduced by ensuring that all metabolic phenotypes that are compatible with the substrate state and the regulatory state are also compatible with the fluxomics observations. This allows identifying which reactions are inactivated due to regulatory network inhibition, and so indicating the regulatory rules that need to be inferred.

Given (v, w, x) a consistent RMSS, we denote by 'phenotype(w, x)' the set of metabolic states compatible with the substrate state w and the regulatory state x. The definition of 'phenotype(w, x)' depends on the formalisms chosen to model the RMN dynamics. Formal definitions according to two RMN dynamics will be provided in Section 3.

▶ **Definition 2.3: Phenotype compatibility**

Let 'growth' be a reaction representing bacteria growth and (v, w, x) be an RMSS. The substrate state w and regulatory state x are phenotype-compatible with fluxomics observations $\hat{v}$ if and only if all the metabolic states they allow are compatible with the observed growth rate $\hat{v}_{\text{growth}}$ (Eq. II.3).

$$\forall\, v' \in \text{phenotype}(w, x), v'_{\text{growth}} = \hat{v}_{\text{growth}} \tag{II.3}$$

This general definition of phenotype compatibility can be extended to handle noise in fluxomics and kinetics observations (see the definition in Section 3).

Example. For example, according to the rFBA dynamics and the growth maximization (Feist and Palsson, 2010), the metabolic phenotype corresponds to the optimal growth allowed by the substrate and regulatory states. The RMSS (v, w, x) is phenotype-compatible with the fluxomics data $\hat{v}$ if the observed growth $\hat{v}_{\text{growth}}$ is equal to the maximum growth allowed by w and x, *i.e.* $\max_{v' \in \text{rMSS}(w,x)} v'_{\text{growth}} = \hat{v}_{\text{growth}}$.

2.3 Compatibility Between Time Series Data and an RMN Traces

Let consider an RMN trace, *i.e.* a sequences of RMSSs, $\mathcal{T}_s = \{(v^i, w^i, x^i)\}_{i=1}^{l}$ of any dynamics, and an observed time series $\mathcal{T}_o = \{(\hat{v}^i, \hat{w}^i, \hat{x}^i)\}_{i=1}^{m}$, with $0 \leq m \leq l$. Given $g : [1; m] \to [1; l]$ a bijective function mapping each observation to an RMSS of $\mathcal{T}_s$, $\mathcal{T}_o$ is compatible with $\mathcal{T}_s$ if each observation $(\hat{v}^i, \hat{w}^i, \hat{x}^i) \in \mathcal{T}_o$ can be associated to an RMSS $(v^{g(i)}, w^{g(i)}, x^{g(i)}) \in \mathcal{T}_s$ such that: **(i)** successive observations are associated with successive RMSSs (Eq. II.4a), **(ii)** the observation and the RMSS are compatible (Eq. II.4b), and **(iii)** the growth phenotype is consistent with the fluxomic observations (Eq. II.4c).

▶ **Definition 2.4: Compatibility with an RMN trace**

An observed time series $\mathcal{T}_o = \{(\hat{v}^i, \hat{w}^i, \hat{x}^i)\}_{i=1}^{m}$ is said *compatible with distance* $0 \leq K$ with an RMN trace $\mathcal{T}_s = \{(v^j, w^j, x^j)\}_{j=1}^{l}$, with $1 \leq m \leq l$, if and only if it exists a bijective function $g : [1; m] \to [1; l]$ mapping observations to RMSSs

such that:

$$\forall\, 1 \leq i < m,$$
$$0 < g(i+1) - g(i) \leq K + 1 \tag{II.4a}$$
$$\wedge\ (v^{g(i)}, w^{g(i)}, x^{g(i)}) \text{ and } (\hat{v}^i, \hat{w}^i, \hat{x}^i) \text{ are data-compatible} \tag{II.4b}$$
$$\wedge\ w^{g(i)} \text{ and } x^{g(i)} \text{ is phenotype-compatible with } \hat{v}^i_{\text{growth}} \tag{II.4c}$$

For the rest, we will say that an RMN trace $\mathcal{T}_s = \{(v^j, w^j, x^j)\}_{j=1}^{l}$ is *exactly compatible* with an observed time series $\mathcal{T}_o = \{(\hat{v}^i, \hat{w}^i, \hat{x}^i)\}_{i=1}^{m}$ if $\mathcal{T}_o$ are compatible with distance $K = 0$ with $\mathcal{T}_s$. In other words, if $\mathcal{T}_o$ and $\mathcal{T}_s$ are of same length.

3 Inference Problem

Inference problem. Let $\mathcal{G}$ be a PKN, $\mathcal{N} = (\mathcal{M}_{\text{ext}} \cup \mathcal{M}_{\text{int}}, \mathcal{R}, s)$ be a metabolic network, and $\mathcal{P}$ be a set of genes and regulatory proteins. Equations II.4 in Section 2.3 characterize the compatibility between a sequence of RMSSs and an observed time series. The problem of inferring BNs controlling a metabolic network from time series data comes down to finding BNs $f \in \mathbb{F}(\mathcal{G})$, supported by the PKN, that best fit the observed time series. The BNs that best fit observed time series are the BNs such that the RMN $(\mathcal{N}, \mathcal{P}, f)$ admits the *minimal length* traces compatible with each observed time series. Indeed, we assume that most of the system states have been observed. Therefore, the inferred BNs' traces, compatible with the observed time series, should minimize the number of RMSSs not associated with observations.

General definition of the inference problem. Let '$dynamics(\mathcal{N}, \mathcal{P}, f)$' be the set of all traces, *i.e.* sequences of RMSSs, compatible with the RMN $(\mathcal{N}, \mathcal{P}, f)$ according to the chosen dynamics, *e.g.* regulated flux balance analysis (rFBA). Let 'phenotype(w, x)' be the set of all metabolic states compatible with a substrate state w and a regulatory state x for the chosen dynamics
Formally, the inference of BNs controlling the metabolism from time series data is defined as the following combinatorial optimization problem under quantified constraints:

General form of the inference problem

Input:

1: a metabolic network $\mathcal{N} = (\mathcal{M}_{\text{ext}} \cup \mathcal{M}_{\text{int}}, \mathcal{R}, s)$ with an objective reaction 'growth';
2: a set of regulatory proteins $\mathcal{P}$;
3: a set of observed time series $\{\mathcal{T}_1, ..., \mathcal{T}_q\}$, $q \geq 1$;
4: a prior knowledge network $\mathcal{G}$ of dimension $n = |\mathcal{P}| + |\mathcal{M}_{\text{ext}}| + |\mathcal{R}|$;
5: a maximum distance $K_{\max} \in \mathbb{N}$ between observations.

Output: $\boldsymbol{\arg\min_{f \in \mathbb{F}(\mathcal{G})} \sum_{k=1}^{q} l_k}$

// The Boolean networks supported by the PKN that best fit the observed time series.

such that:

$$\forall \boldsymbol{\mathcal{T}_k} \in \boldsymbol{\{\mathcal{T}_1, ..., \mathcal{T}_q\}}, \exists \boldsymbol{\{(v^j, w^j, x^j)\}_{j=1}^{l_k} \in \text{dynamics}(\mathcal{N}, \mathcal{P}, f)}, \forall \boldsymbol{1 \leq i < |\mathcal{T}_k|},$$

// For each observed time series $\mathcal{T}_k$, it exists a trace $\{(v^j, w^j, x^j)\}_{j=1}^{l_k}$ of length l_k
// of the RMN dynamics such that:

$$|\mathcal{T}_k| \leq l_k \leq |\mathcal{T}_k| + K_{\max} \tag{II.5a}$$

// At most K_{max} RMSSs of the trace are not associated with observations.

$$\wedge\ 0 < g_k(i+1) - g_k(i) \tag{II.5b}$$

// The order of the observations is kept.

$$\wedge\ (v^{g_k(i)}, w^{g_k(i)}, x^{g_k(i)}) \text{ and } (\hat{v}^i, \hat{w}^i, \hat{x}^i) \text{ are data-compatible} \tag{II.5c}$$

// The observation is data-compatible with its associated RMSS (Def. 2.2).

$$\wedge\ \forall v' \in \text{phenotype}(w^{g_k(i)}, x^{g_k(i)}), v'_{\text{growth}} = \hat{v}^i_{\text{growth}} \tag{II.5d}$$

// The substrate and regulatory states are phenotype-compatible with the
// fluxomics observations (Def. 2.3).

where $g_k : [0, |\mathcal{T}_k|] \to [0, l_k]$ is a bijective function mapping observations of the observed time series $\mathcal{T}_k$ to RMSSs of the trace $\{(v^j, w^j, x^j)\}_{j=1}^{l_k}$.

For the rest, we will denote as the *exact inference problem* the problem of inferring regulatory networks such that it exists, for each observed time series $\mathcal{T}_k$, a trace of the regulated metabolic network $(\mathcal{N}, \mathcal{P}, f)$ *exactly compatible* with $\mathcal{T}_s$, *i.e.* the inference problem where $K_{\max}$ is fixed to 0. Note that, unlike the inference problem, the *exact* inference problem is a satisfiability problem.

Impact of compatibility constraints on the inferred regulatory rules. The data-compatibility (Eq. II.5c) and phenotype-compatibility (Eq. II.5d) constraints are essential to infer all the regulatory rules. Specifically, data-compatibility allows for inferring *'standard' regulatory* rules and *feedback* rules, and phenotype-compatibility allows inferring *control* rules.

The former fixes the states of the regulatory proteins and genes, as well as the external metabolite availability and reaction activity states. In other words, it fixes both the input and output of the 'standard' regulatory and feedback rules.

The latter is necessary for the control rules. Recall that the data-compatibility constraint does not fix the regulatory states of reactions, the regulatory states of inactivated reactions are uncertain. A reaction can be inactivated because of an inhibition by control rules, or due to metabolic-related constraints. The phenotype-compatibility constraint resolves these uncertainties. It ensures that all metabolic phenotypes, compatible with the set of reactions inhibited by control rules, are compatible with the observations. This allows for identifying reactions that should be inhibited, and so, for which control rules should be learned.

Outlines. In the next chapters, we solve the inference problem according to two RMN dynamics: a Boolean abstraction of rFBA (see Chapter III), and rFBA (see Chapters IV and V).

In the next sections, we briefly present three formulations of the inference problem derived from these two dynamics.

3.1 Relaxed Boolean Definition

Definition of the inference problem described and solved in Chapter III.

Boolean abstraction of rFBA. We first propose a relaxation of the exact inference problem as a Boolean satisfiability problem using a Boolean abstraction of the rFBA dynamics. The Boolean rFBA dynamics relies on Boolean metabolic steady-states (BMSS), a Boolean abstraction of the metabolic steady-states. It abstracts the linear constraints of the FBA with logical constraints. A BMSS is represented by a Boolean-valued vector $v \in \mathbb{B}^{|\mathcal{R}|}$. For the rest, the set of all Boolean rFBA traces of a regulated metabolic network $(\mathcal{N}, \mathcal{P}, f)$ is denoted by $\text{rFBA}^{\mathbb{B}}(\mathcal{N}, \mathcal{P}, f)$. BMSSs and the Boolean rFBA dynamics are formally defined in Chapter III.

Phenotype-compatibility. Let $\hat{o} : \mathbb{B} \to \mathbb{N}$ be a Boolean objective function quantifying the growth phenotype. This Boolean objective function replaces the objective reaction used in rFBA. Indeed, a BMSS does not have qualitative information on

reactions, only the state (on/off) of a reaction is known, therefore we could not rely on the metabolic flux on a growth reaction to characterize the bacteria growth.

Let $(\hat{v}, \hat{w}, \hat{x})$ be an observation, and (v, w, x) be a Boolean regulated metabolic steady-states. Given $\text{MSS}^{\mathbb{B}}(\mathcal{N})$ the set of all BMSSs compatible with a metabolic network $\mathcal{N}$, the phenotype-compatibility consists in ensuring that all BMSSs $v' \in \text{MSS}^{\mathbb{B}}$ allowed by the substrate state w and the regulatory state x have an objective score lesser or equals than the objective score of the observation (Eq II.6).

$$\forall\, (v', w', x') \in \text{MSS}^{\mathbb{B}}(\mathcal{N}),\ \ w \neq w' \vee x \neq x' \vee \hat{o}(v') \leq \hat{o}(\hat{v}) \tag{II.6}$$

This definition assumes that bacteria aim at maximizing their growth, and so, that no BMSS should exhibit more growth than the observation.

Relaxation formulation of the inference problem. Based on Boolean rFBA and the phenotype compatibility described by Eqs II.6, the Boolean relaxation of the inference problem as a Boolean *satisfiability* problem is formally given below. Elements in **blue** are the elements that changed from the general definition.

Boolean relaxation of the exact inference problem

Input:

1: a metabolic network $\mathcal{N} = (\mathcal{M}_{\text{ext}} \cup \mathcal{M}_{\text{int}}, \mathcal{R}, s)$;
2: a set of regulatory proteins $\mathcal{P}$;
3: a set of observed time series $\{\mathcal{T}_o^1, \cdots, \mathcal{T}_o^q\}$, $q \geq 1$;
4: a prior knowledge network $\mathcal{G}$ of dimension $n = |\mathcal{P}| + |\mathcal{M}_{\text{ext}}| + |\mathcal{R}|$;
5: **a Boolean objective function $\hat{o} : \mathbb{B}^{|\mathcal{R}|} \to \mathbb{N}$.**

Output: **All Boolean network $f \in \mathbb{F}(\mathcal{G})$**
such that:

$$\forall\, \mathcal{T}_k \in \{\mathcal{T}_1, \cdots, \mathcal{T}_q\},\ \exists\, \{(v^j, w^j, x^j)\}_{j=1}^{l_k} \in \mathbf{rFBA}^{\mathbb{B}}(\boldsymbol{\mathcal{N}}, \boldsymbol{\mathcal{P}}, \boldsymbol{f}),\ \forall 1 \leq i < |\mathcal{T}_k|,$$

$$\boldsymbol{l_k = |\mathcal{T}_k|} \tag{II.7a}$$

$$\wedge\ 0 < g_k(i+1) - g_k(i) \tag{II.7b}$$

$$\wedge\ (v^{g_k(i)}, w^{g_k(i)}, x^{g_k(i)}) \text{ and } (\hat{v}^i, \hat{w}^i, \hat{x}^i) \text{ are data-compatible} \tag{II.7c}$$

$$\wedge\ \forall\, \boldsymbol{(v', w', x') \in \text{MSS}^{\mathbb{B}}(\mathcal{N}),}$$

$$\boldsymbol{w^{g(i)} \neq w' \vee x^{g(i)} \neq x' \vee \hat{o}(v') \leq \hat{o}(\hat{v}^i)} \tag{II.7d}$$

where $g_k : [0, |\mathcal{T}_k|] \to [0, l_k]$ is a bijective function mapping observations of the observed time series $\mathcal{T}_k$ to RMSSs of the trace $\{(v^j, w^j, x^j)\}_{j=1}^{l_k}$.

Solving. The relaxed definition is a Boolean satisfiability problem with two levels of quantifiers (2-QBF) of the form '$\exists\forall$': it exists a Boolean network $f \in \mathbb{F}(\mathcal{G})$ that admits a set of Boolean rFBA traces compatible with the observed time series, such that all Boolean metabolic steady-states satisfied the fluxomics observations (Eq. II.7d). To solve this formulation of the inference problem, we propose an ASP-based implementation based on the so-called saturation technique (Eiter et al., 2009; Gebser et al., 2011), an efficient solving strategy for 2-QBF problems.

3.2 Flux-based Definition

Definition of the inference problem described and solved in Chapter IV.

Based on rFBA. Then, we define the inference problem based on the rFBA dynamics. For the rest, we denote the set of all rFBA traces of a regulated metabolic network $(\mathcal{N}, \mathcal{P}, f)$ by $\text{rFBA}(\mathcal{N}, \mathcal{P}, f)$. Recall that $\text{rMSS}(\mathcal{N}, w, x)$ is the set of RMSSs of $\mathcal{N}$ compatible with the substrate state $w \in \mathbb{R}^{|\mathcal{M}_{\text{ext}}|}$ and the regulatory state $x \in \mathbb{B}^{|\mathcal{P}|+|\mathcal{M}_{\text{ext}}|+|\mathcal{R}|}$.

The phenotype compatibility is based on the FBA assumption that bacteria aim at maximizing their growth (Feist and Palsson, 2010). Hence, the metabolic phenotype is characterized by the optimal growth allowed by a substrate and a regulatory state. An RMSS (v, w, x) is phenotype-compatible with an observation $(\hat{v}, \hat{w}, \hat{x})$ if both the current growth (v_{Growth}) and its optimal growth match with the observed growth. Given a noise rate $0 \leq \epsilon < 1$, the phenotype compatibility is formally defined as:

$$\frac{\hat{v}_{\text{growth}}}{1+\epsilon} \leq v_{\text{growth}} \;\wedge\; \max_{v' \in \text{rMSS}(\mathcal{N},w,x)} v'_{\text{growth}} \leq \frac{\hat{v}_{\text{growth}}}{1-\epsilon} \tag{II.8}$$

where 'growth' is an objective reaction modeling bacteria growth. The noise rate parameter ϵ allows taking into account the noise in the fluxomics data.

Flux-based formulation. The flux-based definition of the inference problem according to the rFBA dynamics is given below. The problem is formulated as a hybrid combinatorial optimization problem under logic and quantified linear constraints. Elements in **blue** are the elements that changed from the general definition.

Flux-based formulation of the inference problem

Input:

1: a metabolic network $\mathcal{N} = (\mathcal{M}_{\text{ext}} \cup \mathcal{M}_{\text{int}}, \mathcal{R}, s)$ with an objective reaction 'growth';
2: a set of regulatory proteins $\mathcal{P}$;
3: a set of observed time series $\{\mathcal{T}_o^1, \cdots, \mathcal{T}_o^q\}$, $q \geq 1$;
4: a prior knowledge network $\mathcal{G}$ of dimension $n = |\mathcal{P}| + |\mathcal{M}_{\text{ext}}| + |\mathcal{R}|$;
5: a maximum distance $K_{\max} \in \mathbb{N}$ between observations;
6: **a noise rate parameter $\epsilon \in [0, 1[$.**

Output: $\underset{f \in \mathbb{F}(\mathcal{G})}{\arg\min} \sum_{k=1}^{q} l_k$

such that:

$$\forall \mathcal{T}_k \in \{\mathcal{T}_1, \cdots, \mathcal{T}_q\},\ \exists \{(v^j, w^j, x^j)\}_{j=1}^{l_k} \in \mathbf{rFBA(\mathcal{N}, \mathcal{P}, f)},\ \forall 1 \leq i < |\mathcal{T}_k|,$$

$$|\mathcal{T}_k| \leq l_k \leq |\mathcal{T}_k| + K_{\max} \tag{II.9a}$$

$$\wedge\ 0 < g_k(i+1) - g_k(i) \tag{II.9b}$$

$$\wedge\ (v^{g_k(i)}, w^{g_k(i)}, x^{g_k(i)}) \text{ and } (\hat{v}^i, \hat{w}^i, \hat{x}^i) \text{ are data-compatible} \tag{II.9c}$$

$$\wedge\ \frac{\hat{\boldsymbol{v}}^{\boldsymbol{i}}_{\text{growth}}}{\mathbf{1} + \boldsymbol{\epsilon}} \leq \boldsymbol{v}^{\boldsymbol{g_k(i)}}_{\text{growth}}\ \wedge\ \max_{\boldsymbol{v' \in rMSS(\mathcal{N}, w^{g_k(i)}, x^{g_k(i)})}} \boldsymbol{v'_{\text{growth}}} \leq \frac{\hat{\boldsymbol{v}}^{\boldsymbol{i}}_{\text{growth}}}{\mathbf{1} - \boldsymbol{\epsilon}} \tag{II.9d}$$

where $g_k : [0, |\mathcal{T}_k|] \to [0, l_k]$ is a bijective function mapping observations of the observed time series $\mathcal{T}_k$ to RMSSs of the trace $\{(v^j, w^j, x^j)\}_{j=1}^{l_k}$.

Solving. This flux-based definition of the inference problem is a hybrid problem merging logical and linear constraints. The linear constraints are the FBA equations used to define RMSS ('rMSS($\mathcal{N}, w, x$)'), and so, to define the rFBA traces ('rFBA($\mathcal{N}, \mathcal{P}, f$)').

To solve this hybrid problem, we introduce *MERRIN*, a dedicated hybrid solving framework that allows solving the logical constraints with the FBA constraints.

3.3 Optimization Modulo Quantified Linear Arithmetic Definition

Definition of the inference problem, and the class of problems described and solved in Chapter V.

Generalization as an OPT+qLP problem. Finally, we generalize the dedicated solving method used to address the flux-based inference problem to a broader class of hybrid problems: optimization modulo quantified linear arithmetic problems (OPT+qLP). An OPT+qLP problem is a hybrid optimization problem that merges logic and quantified linear constraints. We assume that quantifiers over the linear constraints are either the universal quantifier ($\forall$) or the existential quantifier ($\exists$), and that there is only one level of quantifier over linear constraints.

For the flux-based formulation, we define the phenotype compatibility as a constraint over the maximum growth allowed by a substrate state w and a regulatory state x (Eq. II.8). In practice, ensuring the lower bound of the constraint ($\frac{\hat{v}_{\text{growth}}}{1+\epsilon} \leq v_{\text{growth}}$) does not change (Eq. II.10a). Ensuring the upper bound ($\max_{v' \in \text{rMSS}(w,x)} v'_{\text{growth}} \leq \frac{\hat{v}_{\text{growth}}}{1-\epsilon}$) is equivalent to ensuring that all RMSS compatible with w and x exhibit a growth phenotype lesser or equal to the observed one (Eq. II.10b). Therefore, equation II.8 can be converted into an equivalent set of existentially and universally quantified linear constraints (Eqs. II.10).

$$\frac{\hat{v}_{\text{Growth}}}{1+\epsilon} \leq v_{\text{Growth}} \tag{II.10a}$$

$$\wedge\ \forall v \in \text{rMSS}(w,x),\ v'_{\text{Growth}} \leq \frac{\hat{v}_{\text{Growth}}}{1-\epsilon} \tag{II.10b}$$

OPT+qLP definition. The OPT+qLP definition of the inference problem according to the rFBA dynamics is given below. Elements in **blue** are the elements that changed from the general definition. It differs from the flux-based definition (Section 3.2) on Eqs. II.11d and II.11e, where the phenotype compatibility is reformulated with existentially and universally quantified linear constraints.

OPT+qLP formulation of the inference problem

Input:

1: a metabolic network $\mathcal{N} = (\mathcal{M}_{\text{ext}} \cup \mathcal{M}_{\text{int}}, \mathcal{R}, s)$ with an objective reaction 'growth';
2: a set of regulatory proteins $\mathcal{P}$;
3: a set of observed time series $\{\mathcal{T}_o^1, \cdots, \mathcal{T}_o^q\}$, $q \geq 1$;
4: a prior knowledge network $\mathcal{G}$ of dimension $n = |\mathcal{P}| + |\mathcal{M}_{\text{ext}}| + |\mathcal{R}|$;
5: a maximum distance $K_{\max} \in \mathbb{N}$ between observations;
6: **a noise rate parameter $\epsilon \in [0, 1[$.**

Output: $\underset{f \in \mathbb{F}(\mathcal{G})}{\arg\min} \sum_{k=1}^{q} l_k$
such that:

$$\forall \mathcal{T}_k \in \{\mathcal{T}_1, \cdots, \mathcal{T}_q\},\ \exists \{(v^j, w^j, x^j)\}_{j=1}^{l_k} \in \mathbf{rFBA(\mathcal{N}, \mathcal{P}, f)},\ \forall 1 \leq i < |\mathcal{T}_k|,$$

$$|\mathcal{T}_k| \leq l_k \leq |\mathcal{T}_k| + K_{\max} \tag{II.11a}$$

$$\wedge\ 0 < g_k(i+1) - g_k(i) \tag{II.11b}$$

$$\wedge\ (v^{g_k(i)}, w^{g_k(i)}, x^{g_k(i)}) \text{ and } (\hat{v}^i, \hat{w}^i, \hat{x}^i) \text{ are data-compatible} \tag{II.11c}$$

$$\wedge\ \frac{\hat{\boldsymbol{v}}^{\boldsymbol{i}}_{\mathbf{growth}}}{\mathbf{1} + \boldsymbol{\epsilon}} \leq \boldsymbol{v}^{\boldsymbol{g_k(i)}}_{\mathbf{growth}} \tag{II.11d}$$

$$\wedge\ \forall \boldsymbol{v'} \in \mathbf{rMSS(\mathcal{N}, w^{g_k(i)}, x^{g_k(i)})},\ \boldsymbol{v'}_{\mathbf{growth}} \leq \frac{\hat{\boldsymbol{v}}^{\boldsymbol{i}}_{\mathbf{growth}}}{\mathbf{1} - \boldsymbol{\epsilon}} \tag{II.11e}$$

where $g_k : [0, |\mathcal{T}_k|] \to [0, l_k]$ is a bijective function mapping observations of the observed time series $\mathcal{T}_k$ to RMSSs of the trace $\{(v^j, w^j, x^j)\}_{j=1}^{l_k}$.

Solving. In chapter V, we introduce a novel generic solving framework to solve any OPT+qLP problems. This framework is based on a generalization of the dedicated solving framework developed to solve the flux-based formulation. It falls within the so-called *Counter-Example Guided Abstraction Refinement* method (Clarke et al., 2003), already used to solve 2-QBF and Satisfiability Modulo Theory problems (Janota et al., 2016; Brummayer and Biere, 2009; Lagniez et al., 2017).

In next chapters

The next three chapters introduce the different contributions of the thesis. First, chapter III presents the Boolean relaxation of the inference problem with the saturation-based method used to solve it. Then, chapter IV presents the flux-based formulation of the inference problem and the hybrid solving methods that we developed to solve it. Finally, chapter V generalizes the solving framework presented in chapter IV to a broader class of hybrid problems, namely OPT+qLP problems, of which the flux-based formulation of the inference problem is an example.

Chapters structure. Each chapter is based on a published paper and follows the same structure:

1. The reference of the associated paper and a summary of the chapter's contributions and results.
2. A preliminary section that situates and motivates the paper's contributions within the thesis subject. It presents the formulation of the inference problem solved in the chapter, and how the formulation and proposed solving method overcome the limitation of previous work.
3. A highlight of the paper's main contributions regarding the inference problem. This section goes beyond the paper's contributions, we present the different challenges that we face and the methods used to overcome them.
4. A discussion about current results and limitations regarding the inference problem. This discussion goes beyond the paper's discussion, we introduce new results and discuss in further detail the current limitations.
5. The paper associated with the chapter.

III Boolean Abstraction of rFBA for the Boolean Relaxation of the Exact Inference Problem

In this chapter, we introduce a Boolean abstraction of the regulatory flux balance analysis (rFBA) framework to relax the exact inference problem as a combinatorial satisfiability problem with two levels of quantifiers. The content of this chapter has been presented at the international conference on *Computational Methods in Systems Biology* (CMSB) of 2021 with the associated paper published in the conference proceedings (Thuillier et al., 2021).

To sum up

We introduce a first method to infer Boolean regulatory rules from time series observations of the metabolic and regulatory scales. This method is based on a Boolean abstraction of the rFBA framework, enabling the formulation of the exact inference problem as a Boolean satisfiability problem with two levels of quantifiers (2-QBF). The Boolean relaxation of this inference problem has been tackled using Answer Set Programming (ASP) with the so-called saturation method. However, due to the Boolean abstractions, the results are highly dependent on the input Boolean objective function, which may lead to the inference of false positive regulatory networks. This limitation necessitates further refinement to improve the reliability of the inferred networks.

In this chapter

1 Problem Statement

As far as we know, there are no methods to infer Boolean networks (BN) controlling the metabolism. Existing BNs inferring methods only account for the discrete dynamics of the regulatory scale and rely on time series of transcriptomics or phosphoproteomics data (Videla et al., 2017; Chevalier et al., 2020). They model the inference problem as a combinatorial satisfiability, or optimization, problem.

Boolean relaxation. Following on from these methods, we propose a Boolean relaxation of the inference problem as a pure Boolean satisfiability problem. This Boolean relaxation is based on a Boolean abstraction of rFBA, based on a discrete approximation of the metabolic fluxes, that is, of the FBA equations. A Boolean abstraction of the metabolic steady-state is called a Boolean metabolic steady-state (BMSS) and is represented by a Boolean-valued vector $v \in \mathbb{B}^{|\mathcal{R}|}$, where $\mathcal{R}$ is a set of reactions. BMSSs and the Boolean abstraction of rFBA are formally defined in `Section 3` of the paper.

For the rest, we will denote by $\text{MSS}^{\mathbb{B}}(\mathcal{N})$ the set of all BMSSs compatible with a metabolic network $\mathcal{N}$, and by $\text{rFBA}^{\mathbb{B}}(\mathcal{N}, \mathcal{P}, f)$ the set of all the traces of a regulated metabolic network (RMN) $(\mathcal{N}, \mathcal{P}, f)$ compatible with our Boolean abstraction of rFBA. The Boolean relaxation of the exact inference problem is defined as:

Boolean relaxation of the exact inference problem

Input:

1: a metabolic network $\mathcal{N} = (\mathcal{M}_{\text{ext}} \cup \mathcal{M}_{\text{int}}, \mathcal{R}, s)$;
2: a set of regulatory proteins $\mathcal{P}$;
3: a set of observed time series $\{\mathcal{T}_o^1, \cdots, \mathcal{T}_o^q\}$, $q \geq 1$;
4: a prior knowledge network $\mathcal{G}$ of dimension $n = |\mathcal{P}| + |\mathcal{M}_{\text{ext}}| + |\mathcal{R}|$;
5: a Boolean objective function $\hat{o} : \mathbb{B}^{|\mathcal{R}|} \to \mathbb{N}$.

Output: All Boolean network $f \in \mathbb{F}(\mathcal{G})$
such that:

$$\forall\, \mathcal{T}_k \in \{\mathcal{T}_1, \cdots, \mathcal{T}_q\},\ \exists\, \{(v^j, w^j, x^j)\}_{j=1}^{l_k} \in \text{rFBA}^{\mathbb{B}}(\mathcal{N}, \mathcal{P}, f),\ \forall\, 1 \leq i < |\mathcal{T}_k|,$$

$$l_k = |\mathcal{T}_k| \tag{III.1a}$$

$$\wedge\ 0 < g_k(i+1) - g_k(i) \tag{III.1b}$$

$$\wedge\ (v^{g_k(i)}, w^{g_k(i)}, x^{g_k(i)}) \text{ and } (\hat{v}^i, \hat{w}^i, \hat{x}^i) \text{ are data-compatible} \tag{III.1c}$$

$$\wedge\ \forall\, (v', w', x') \in \text{MSS}^{\mathbb{B}}(\mathcal{N}),$$

$$w^{g(i)} \neq w' \vee x^{g(i)} \neq x' \vee \hat{o}(v') \leq \hat{o}(\hat{v}^i) \tag{III.1d}$$

where $g_k : [0, |\mathcal{T}_k|] \to [0, l_k]$ is a bijective function mapping observations of the observed time series $\mathcal{T}_k$ to RMSSs of the trace $\{(v^j, w^j, x^j)\}_{j=1}^{l_k}$.

Outlines. In this chapter, we briefly present the two main contributions of the paper: the Boolean abstraction of metabolic steady-states (Section 2.1), which forms the foundation of all the inference methods presented in this manuscript, and the saturation method used to solve the Boolean relaxation of the exact inference problem (Section 2.2). Finally, results from another case study are provided in Section 3, along with a discussion on the limitations of the Boolean abstraction of metabolic steady-states.

2 Contributions of the CMSB's paper

2.1 Boolean abstraction of metabolic steady-state

Overview of the Boolean abstraction of metabolic steady-state defined in `Section 3.1` *of the paper.*

To model the exact inference problem as a pure Boolean satisfiability problem, we must abstract the linear dynamics of the metabolism, and thus the rFBA equations.

Exact Boolean abstraction. Ideally, we would like the Boolean abstraction of metabolic steady-states to be exact, that is, the set of all BMSSs to be equal to the set of metabolic steady-state supports. In other words, we seek an exact Boolean abstraction of the linear systems modeled by the rFBA equations. While it is theoretically possible to define exact Boolean abstractions of linear systems (Allart et al., 2021), these abstractions require computing all the minimal generator vectors of the linear systems. For metabolic networks, it comes down to computing all elementary flux modes (EFMs) (Schuster and Hilgetag, 1994). Currently, there is no efficient and scalable way to enumerate these vectors for genome-scale metabolic networks (Ullah et al., 2019). Therefore, in practice, an exact Boolean abstraction of metabolic steady-states is not feasible.

> ***Challenge***
> Defining a Boolean over-approximation of metabolic steady-states. The abstraction should be usable in place of the metabolic steady-states in the rFBA formalism.

Boolean metabolic steady-states. Only the activation states of reactions are needed to infer regulatory rules. Given $\mathcal{N} = (\mathcal{M}_{\text{ext}} \cup \mathcal{M}_{\text{int}}, \mathcal{R}, s)$ a metabolic network, we derive a logical characterization of the notion of steady-state, considering that a reaction is either active or inactive.

Let $\bar{v} \in \mathbb{R}^{|\mathcal{R}|}$ be a Boolean-valued vector that models the activity state of each reaction $r \in \mathcal{R}$, with $v_r = 1$ if and only if r is active. In the FBA equation, a metabolic state $v \in \mathbb{R}^{|\mathcal{R}|}$ is at steady-state if the sum of input metabolic fluxes is equal to the sum of output metabolic fluxes for each internal metabolite (Eq. III.2).

$$\forall\, m \in \mathcal{M}_{\text{int}},\ \sum_{\substack{r \in \mathcal{R} \\ s_{mr} < 0}} s_{mr} \times v_r = \sum_{\substack{r \in \mathcal{R} \\ s_{mr} > 0}} s_{mr} \times v_r \tag{III.2}$$

In the same way, the Boolean vector $\bar{v}$ is at steady-state if and only if each internal metabolite that is produced (*resp.* consumed) by an active reaction is also consumed (*resp.* produced) by another active reaction (Eq. III.3).

$$\forall\, m \in \mathcal{M}_{\text{int}},\ \bigvee_{\substack{r \in \mathcal{R} \\ s_{mr} < 0}} \bar{v}_r \iff \bigvee_{\substack{r \in \mathcal{R} \\ s_{mr} > 0}} \bar{v}_r \tag{III.3}$$

Boolean vectors satisfying Eq. III.3 are Boolean metabolic steady-states (BMSS).

$$\begin{aligned} R_1 &: \quad A \to B \\ R_2 &: \quad 2 \cdot B \to A \end{aligned}$$

(a) Reactions

$$\begin{aligned} A &: \quad v_{R_1} = v_{R_2} \\ B &: \quad v_{R_1} = 2 \times v_{R_2} \end{aligned}$$

(b) Steady-state equations

$$\begin{aligned} A &: \quad \bar{v}_{R_1} \iff \bar{v}_{R_2} \\ B &: \quad \bar{v}_{R_1} \iff \bar{v}_{R_2} \end{aligned}$$

(c) Boolean steady-state equations

Figure 15 – Example of the steady-state and the Boolean steady-state equations for two reactions R_1 and R_2. Figure **(a)** describes the two reactions. Figures **(b)** and **(c)** are the steady-state and Boolean steady-state equations associated with R_1 and R_2, respectively.

For the rest, we will denote by $\text{MSS}^{\mathbb{B}}(\mathcal{N})$ the set of all Boolean metabolic steady-states of the metabolic network $\mathcal{N}$.

Over-approximation. The BMSSs are an over-approximation of metabolic steady-states (MSSs). For each regulated metabolic steady-state (v, w, x) of the RMN $(\mathcal{N}, \mathcal{P}, f)$, the support of the metabolic state v, denoted by $\beta(v)$, is a BMSS, *i.e.* $\beta(v) \in \text{MSS}^{\mathbb{B}}(\mathcal{N})$. The corollary is not true, not all BMSSs can be associated with a metabolic steady-state. Since the logical characterization of metabolic steady-states neglects stoichiometry, BMSSs may have no real-valued counterpart. A BMSS $\bar{v}$ is said *spurious* if there is no metabolic steady-state v whose support $\beta(v)$ is equal to $\bar{v}$.

Example. For instance, let us consider the two reactions R_1 and R_2 described in Fig. 15(a): $R_1 : A \to B$ and $R_2 : 2 \cdot B \to A$. Figure 15(b) gives the steady-state equations associated with R_1 and R_2. They admit only one solution: $v_A = v_B = 0$. Figure 15(c) gives the Boolean steady-state equations associated with R_1 and R_2. They are BMSSs satisfying them: $\bar{v}_{R_1} = \bar{v}_{R_2} = 0$ and $\bar{v}_{R_1} = \bar{v}_{R_2} = 1$. The former BMSS corresponds to the only MSS compatible with steady-state equations. The latter is a spurious BMSS.

***Solution* – in short**
We introduce a logical characterization of metabolic steady-states, which allows for defining Boolean metabolic steady-states (BMSS). The set of all BMSSs of a metabolic network is an over-approximation of the set of all metabolic steady-states' supports.

This Boolean abstraction of metabolic steady-states is the foundation of all the inferring methods introduced in this manuscript. In Chapter IV, we introduce a hybrid inferring method to address the flux-based inference problem. This method

is based on the solving and refinement of an over-approximation of the inference problem built on BMSS.

Boolean abstraction of rFBA. From the definition of Boolean metabolic steady-states, we build a Boolean abstraction of rFBA. Due to the Boolean abstraction, there is no quantitative information on reaction activities, meaning that the metabolic flux passing through a growth reaction cannot be used to rank BMSS; the growth reaction will simply be active or inactive. To address this issue, we introduce a *Boolean objective function* $\hat{o} : \mathbb{B}^{|\mathcal{R}|} \to \mathbb{N}$ to rank BMSSs. Only the BMSS maximizing the Boolean objective function will be selected by the Boolean rFBA dynamics. In practice, the objective function is highly dependent on the metabolic network and PKN structure, it is defined as an input to the relaxed inference problem.

2.2 Saturation-based Solving Framework

Overview of the saturation-based method described in `Section 4.2` *of the paper.*

2-QBF problem. The Boolean relaxation of the exact inference problem is a satisfiability problem with two levels of Boolean quantifiers (2-QBF). It is of the form '$\exists\,\forall$': ($\exists$) it exists a Boolean network $f \in \mathbb{F}(\mathcal{G})$ that admits a set of Boolean rFBA traces compatible with the observed time series, such that ($\forall$) all BMSSs satisfied the maximal growth observations. The 2-QBF problems are known to be Σ_{P}^2-complete (Eiter and Gottlob, 1995).

> ***Challenge***
> Identifying efficient encoding to address 2-QBF problems.

In Answer Set Programming (ASP), the *saturation technique* is an efficient way to encode 2-QBF problems (Gebser et al., 2011). Saturation relies on the stable model semantics of ASP and disjunctive logic program to explore the set of all feasible solutions, and so ensure universally quantified constraints.

Disjunctive logic program. ASP allows for defining *disjunctive logic program* by adding disjunctive declaration in the rule head[1] (Lobo et al., 1992). If the rule body holds, then at least one atom of the rule head should hold. In ASP, disjunctive rules are of the following form:

$$a_0; \cdots; a_m : -\text{body}$$

[1] ASP rules are of the form `head :- body.` (see Chapter I Section 3.1.1 for details).

To solve disjunctive rules, ASP relies on *subset-minimal semantics* (Eiter et al., 2009). Only the subset-minimal models of the disjunctive rules are kept. An answer set containing a set of disjunctive variables $\mathbb{S}$ is considered a solution if and only if there is no other solution answer sets where the set of disjunctive variables is a subset of $\mathbb{S}$. For instance, let's consider the following example:

$$a \text{ :- } .$$
$$b \text{ :- } a.$$
$$a; b; c \text{ :- } .$$

where a, b and c are atoms. There are two solution answer sets: $\{a, b\}$ and $\{a, b, c\}$. However, the atoms a, b and c are declared with a disjunctive rule. As the solution $\{a, b, c\}$ is not subset minimal (it contains $\{a, b\}$), it is discarded. The only subset minimal solution of this disjunctive logic program is $\{a, b\}$.

Saturation. The *saturation technique* (Gebser et al., 2011) allows for encoding and solving 2-QBF problems ('$\exists\,\forall$') with ASP. Let $\exists\, x \forall\, y, \phi(x, y)$ be a 2-QBF problem defined such that given $x \in \mathbb{B}$ all possible assignments of $y \in \mathbb{B}^n$ should satisfy a condition ϕ. The set of universally quantified variables is defined with a disjunctive rule:

$$y_1; \cdots; y_n \text{ :- } x.$$

The saturation technique consists of saturating the set of disjunctive variables y that satisfy a condition ϕ. If a subset of the disjunctive variables satisfies ϕ, then all the disjunctive variables are added to the answer set, $\forall\, 1 \leq i \leq n$ there is:

$$y_i \text{ :- } \phi.$$

Therefore, all valid assignments of the disjunctive variables satisfying ϕ will be saturated. Since disjunctive logic programs follow the subset-minimal semantics, the ASP solver tries to find unsaturated sets of disjunctive constraints, *i.e.* sets of disjunctive constraints that do not satisfy ϕ. In this way, the ASP solver iterates over all possible assignments of disjunctive variables. By prohibiting ϕ to be not satisfied with an integrity constraint (: −not ϕ.), we ensure that the condition ϕ holds for all possible assignment.

***Solution* – in short**
Using a saturation-based encoding, we propose an ASP program to address the Boolean relaxation of the exact inference problem. The saturation method relies on the stable semantics of ASP and disjunctive programming for efficient encoding and solving of 2-QBF problems.

The ASP encoding of the relaxed inference problem and the ASP program, based on the saturation encoding, used to address it are provided in Appendices B.1 and B.2, respectively.

3 Complementary Benchmarking and Discussion

This section extends the discussion of the paper. We present a second application of the relaxed inference problem on a model of core-carbon metabolism.

3.1 Summary of CMSB's Paper

In this chapter, we propose a relaxation of the exact inference problem as a pure Boolean satisfiability problem with two levels of quantifiers (2-QBF). This Boolean relaxation is based on an over-approximation of metabolic steady-states allowing defining a Boolean abstraction of rFBA.

Based on the saturation technique, we introduce an ASP encoding for the relaxed inference problem that we apply to two case-study regulated metabolic models: a *toy* model (see the associated paper), and a model of *core*-carbon metabolism (see Section 3.2). The results obtained from the two case studies are promising. Despite being a relaxation of the inference problem, most of the inferred BNs have been able to reproduce exactly the rFBA simulations used to generate the input time series. However, it is worth noting that false positive BNs are inferred, *i.e.* for which the Boolean rFBA traces are based on spurious BMSSs; and that results are highly dependent on the input Boolean objective function.

Among the results introduced in this chapter, the Boolean abstraction of metabolic steady-states lays the foundation for defining more precise and efficient inferring methods. In particular, this abstraction is the basis of the hybrid inferring framework introduced in the next chapter (Chapter IV).

3.2 Application on a Core-Carbon Metabolic Model.

The instance and results analysis are described in detail in Appendix C.

Core-carbon metabolic model. While not part of the publication, we apply the relaxed inference problem on the model of core-carbon metabolism introduced in (Covert et al., 2001). The case-study model (*toy* model), introduced in the paper, is a simplified version of this model. Unlike the *toy* model, the model of core-carbon metabolism (*core* model) contains metabolic cycles and feedback regulatory rules. In particular, due to the Boolean abstraction of metabolic steady-states, metabolic cycles are sources of spurious BMSS.

Instance description. The observed time series have been generated from discretized rFBA simulations made from five experimental conditions provided in the model introductory paper. The prior knowledge network has been created from the Boolean regulatory network influence graph by removing all influence signs and directions.

Results. From this instance, 7 680 BNs were inferred, of which 2 are subset minimal. Among the two subset-minimal networks, only one can exactly reproduce the rFBA simulations used to generate the observed time series. The second one is a false positive BN, for which the Boolean rFBA traces (exactly compatible with the observed time series) contain spurious BMSS.

3.3 Limitation: Spurious Boolean Metabolic Steady-States

False positive Boolean networks. Addressing the inference problem using a Boolean relaxation of rFBA has shown promising results. It is the first step toward the solving of the inference of BN controlling the metabolism from time series data. However, it is worth noting that false positive BNs are inferred, *i.e.* BNs that could not reproduce the input rFBA time series. On the core model, half of the BNs inferred were false positives.

Limitations of the BMSS abstraction. The inferring of false positive BNs is due to the Boolean abstraction of metabolic steady-states. It abstracts the quantitative metabolic fluxes on reactions by a qualitative state, active or inactive. This abstraction has two main drawbacks: **(i)** there is no consideration of stoichiometry, and **(ii)** we cannot rely on the maximization of metabolic fluxes to model the growth phenotype.

Currently, the impact of spurious BMSSs on the inferred BNs can be mitigated by the Boolean objective function, which makes the set of inferred BNs highly dependent on the chosen objective function. The Boolean objective function should be defined such that, for each observation and potential candidate BN, no optimal BMSS is spurious. Therefore, finding a Boolean objective function is difficult, and requires high expertise in the metabolic network and the influences between regulatory components.

Learning Boolean controls in regulated metabolic networks: a case-study

Kerian Thuillier[1], Caroline Baroukh[2], Alexander Bockmayr[3], Ludovic Cottret[2], Loïc Paulevé[4], and Anne Siegel[1]

[1] Univ Rennes, Inria, CNRS, IRISA, F-35000 Rennes, France
[2] LIPME, INRAE, CNRS, Université de Toulouse, Castanet-Tolosan, France
[3] Freie Universität Berlin, Institute of Mathematics, D-14195 Berlin, Germany
[4] Univ. Bordeaux, Bordeaux INP, CNRS, LaBRI, UMR5800, F-33400 Talence, France

Abstract. Many techniques have been developed to infer Boolean regulations from a prior knowledge network and experimental data. Existing methods are able to reverse-engineer Boolean regulations for transcriptional and signaling networks, but they fail to infer regulations that control metabolic networks. This paper provides a formalisation of the inference of regulations for metabolic networks as a satisfiability problem with two levels of quantifiers, and introduces a method based on Answer Set Programming to solve this problem on a small-scale example.

Keywords: Inference · Regulated metabolism · Satisfiability problem.

1 Introduction

During the last twenty years, both the amount and the type of available data have allowed scientists to consider intracellular processes as a whole. Boolean networks have been refined to include non-deterministic dynamics in order to model the response of regulatory interactions [16,2,5]. Similarly, the study of metabolism at steady state has led to various constraint-based approaches [19,17], which usually assume that internal metabolites are in a quasi-steady-state (QSS). The classical approach to analyze metabolic networks at steady state is flux balance analysis (FBA) [19]. In this approach, a linear function, e.g. biomass production, is optimized with respect to stoichiometric and thermodynamic constraints, resulting in a linear programming problem (LP).

However, both the Boolean approach for regulation and the QSS approximation for metabolism are often developed *"in solo"*, without considering that cellular biology is multi-layered in the sense that the metabolic layer interacts through feed-forward and feedback loops with the regulatory layer [4,27,21,9]. Indeed, cellular metabolism transforms nutrients into biomass constituents. Metabolic reactions are catalysed by enzymes, which themselves are controlled by a cascade of regulations involving other proteins, metabolites and abiotic factors, such as temperature and pH. A biological system thus has several layers of control, which mutually depend on each other. It cannot be simply viewed as a

purely hierarchical system because there are regulatory feed-forward and feedback mechanisms to inform each layer on the state of the other ones. In concrete terms, some compounds produced by the metabolic layer have the capability to block or induce signaling regulation cascades, which themselves can block or induce transcription of genes leading to changes in the control of the initial metabolic process.

To figure out how gene expression triggers specific phenotypes depending on the environmental constraints [3], several constraint-based approaches for integrating metabolic and regulatory networks have been developed that combine Boolean dynamics for the regulatory layer with quasi-steady-state approximations of the metabolic layer (see [17] for an overview), one of them being FlexFlux [18], which implements the rFBA framework [9]. A major limitation when using such frameworks to analyse regulated metabolic models is that they require a precise description of the regulatory and signaling layers in the form of Boolean rules. A noticeable exception is [24], where RBA is used to deduce regulations according to perturbations of the environment. However, to induce regulations, the authors assume that no feedback from metabolism to regulation occurs, which does not correspond to the functioning of most systems. In practice, these rules are manually curated from the literature or experimental data. This has been done for example in the case of *E. coli* [8,7] and a few other organisms. But, the need for a manual curation of Boolean rules of regulated metabolism is a strong limitation to the use of these frameworks.

Signaling and regulatory rules can be identified from transcriptomic or phosphoproteomics data by solving combinatorial or MILP problems in order to optimize data-fitting and parsimony hypotheses [23,20,26,22,25].

In this direction, the caspoTS and the BoNesis approaches [22,20,26,6] were developed for inferring Boolean rules to model the response of regulatory and signaling networks from multiple time-series data. The goal of this paper is to lay foundation for the extension of these approaches to the inference of regulatory rules driving metabolism. This is done by discretizing both the rFBA framework (especially the QSS approximation) and the metabolic data used as input of the inference procedure.

This paper is structured as follows. Sect. 2 gives the background on the dynamic rFBA framework for the simulation of coupled metabolic and regulatory networks. In Sect. 3, we define a formal Boolean abstraction of dynamic rFBA simulations. Then, in Sect. 4, we build on this Boolean abstraction to express the inference of the logic of metabolic regulations as a satisfiability problem. Finally, in Sect. 5, we apply the obtained inference framework on a case study of simplified core carbon metabolism.

Notations The cardinality of a finite set X is denoted by $|X|$. Given a vector $x \in D^n$ and a set of indices $I \subseteq \{1, \cdots, n\}$, x_I denotes the vector of dimension $|I|$ equal to $(x_i)_{i \in I}$. The Boolean domain is denoted by $\mathbb{B} = \{0, 1\}$. Given two Boolean vectors $x, y \in \mathbb{B}^n$, we write $x \preceq y$ iff $\forall i \in \{1, \cdots, n\}, x_i \leq y_i$. Finally, given a non-negative real vector $s \in \mathbb{R}^n_{\geq 0}$, we denote by $\beta(s) \in \mathbb{B}^n$ its *binarization*, *i.e.* $\forall i \in \{1, \ldots, n\}, \beta(s)_i = 1$, if $s_i > 0$, and $\beta(s)_i = 0$, if $s_i = 0$.

2 Background: regulated metabolic networks

2.1 Coupling metabolic and regulatory networks

A *regulated metabolic network* consists of two layers. The regulatory layer is modeled by a Boolean network, which controls the metabolites and fluxes of the metabolic layer, which is characterized by linear equations. Feedbacks are provided by the components of the metabolic network, which are involved in the Boolean functions associated with the regulatory layer.

Formally, a metabolic network is given by a set of biochemical reactions linked together by the metabolites that they consume and produce.

Definition 1. *A* metabolic network *is a tuple* $\mathcal{N} = (\mathrm{Int}, \mathrm{Ext}, \mathcal{R}, S)$ *with a set of internal metabolites* Int*, a set of external metabolites* Ext*, a set* $\mathcal{R}$ *of irreversible reactions, and a stoichiometric matrix* $S \in \mathbb{R}^{(|\mathrm{Int}|+|\mathrm{Ext}|)\times|\mathcal{R}|}$.

Given flux bounds $l_r, u_r \in \mathbb{R}, 0 \leq l_r \leq u_r$*, for each* $r \in \mathcal{R}$*, a* metabolic steady state *is a flux vector* $v \in \mathbb{R}^{|\mathcal{R}|}$ *with* $S_{\mathrm{Int},\mathcal{R}} \cdot v = 0$ *and* $l_r \leq v_r \leq u_r$*, for all* $r \in \mathcal{R}$*. Here* $S_{\mathrm{Int},\mathcal{R}}$ *denotes the submatrix of* S *whose rows correspond to the internal metabolites.*

For the sake of simplicity, we assume that all reactions are irreversible. Reversible reactions may be split into a forward and backward reaction if necessary.

Definition 2 (Input and output metabolites). *For an external metabolite* $m \in \mathrm{Ext}$*, we denote by* $w_m = w_m(t) \in \mathbb{R}_{\geq 0}$ *the concentration of* m *at time* $t \geq 0$*.*

An external metabolite $m \in \mathrm{Ext}$ *is called an* input (resp. output) metabolite *if there exists a reaction* $r \in \mathcal{R}$ *with* $S_{mr} < 0$ *(resp.* $S_{mr} > 0$*). Here* S_{mr} *denotes the stoichiometric coefficient of metabolite* m *in reaction* r*. The set of all input metabolites is denoted by* $\mathrm{Inp} \subseteq \mathrm{Ext}$*.*

A regulatory network is a set of biological entities (e.g. genes, reactions, metabolites) or even abiotic entities (e.g. temperature, pH) that are linked by causal effects: the activity of some nodes can affect positively or negatively the activity of other nodes. This activity can be represented by a Boolean network.

Definition 3. *A* Boolean network *(BN) of dimension* n *is a function* $f : \mathbb{B}^n \to \mathbb{B}^n$*. For each* $i \in \{1, \ldots, n\}$*, the* i*-th component* $f_i : \mathbb{B}^n \to \mathbb{B}$ *is called the* local function *of* i*.*

The influence graph $G(f)$ *of* f *is a signed digraph* (V, E) *with* $V = \{1, \ldots, n\}$ *and* $E \subseteq V \times \{-, +\} \times V$ *such that* $(i, s, j) \in E$ *if and only if there exists* $x \in \mathbb{B}^n$ *with* $x_i = 0$ *such that* $s \cdot f_j(x) < s \cdot f_j(x_1, \cdots, x_{i-1}, 1, x_{i+1}, \cdots, x_n)$*. In the following we will slightly abuse notation by identifying* $G(f)$ *with its edge set,* i.e. $G(f) = E$*.*

A BN f *is* locally monotone *whenever for each influence* $(i, s, j) \in G(f)$*, there is no influence with opposite sign,* i.e. $(i, -s, j) \notin G(f)$*.*

We assume here that the fluxes of a metabolic network can be controlled by the activity of the input metabolites and additional regulatory proteins. More

precisely, the activity of some reactions can be blocked (forced to have a zero flux) whenever certain conditions on the activity of input metabolites and regulatory proteins are met. Moreover, we assume that the activity of regulatory proteins is mediated by the metabolic network only. The resulting model is then supposed to run on two time scales: the metabolic network is a fast system, which, depending on the activity of input metabolites and regulatory proteins will converge to a steady state of the reactions fluxes; the regulatory network is a slow system, which gets updated once the metabolic network is in steady state.

Definition 4 (Regulated metabolic network). *A* regulated metabolic network *is a triplet* $(\mathcal{N}, \mathcal{P}, f)$ *composed of:*

- *a metabolic network* $\mathcal{N} = (\mathrm{Int}, \mathrm{Ext}, \mathcal{R}, S)$ *with* k *input metabolites* $\mathrm{Inp} = \{e_1, \cdots, e_k\} \subseteq \mathrm{Ext}$ *and* m *reactions* $\mathcal{R} = \{r_1, \cdots, r_m\}$*;*
- *a set of* d *regulatory proteins* $\mathcal{P} = \{p_1, \ldots, p_d\}$
- *a BN* f *of dimension* $n = |\mathrm{Inp}| + |\mathcal{R}| + |\mathcal{P}|$ *where* $\{1, \ldots, n\} = \mathrm{Inp} \cup \mathcal{R} \cup \mathcal{P}$ *such that* $G(f)$ *is a bipartite graph between* $\mathcal{P}$ *and* $\mathrm{Inp} \cup \mathcal{R}$*.*

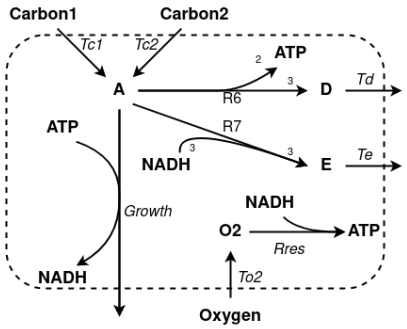
In this work, local functions for input metabolites in the BN f are never used (although the local functions of reactions may depend on them). Therefore we set arbitrarily $f_e = 0, \forall e \in \mathrm{Inp}$.

The BN f models the regulation of the fluxes in the metabolic network $\mathcal{N}$. This regulation is always in one direction: either a flux v_r is only restricted by the flux bounds $l_r \leq v_r \leq u_r$, whenever $f_r(x) = 1$, or it is blocked, $v_r = 0$, whenever $f_r(x) = 0$. Following this convention, a reaction $r \in \mathcal{R}$ is never regulated whenever $f_r(x) = 1$. As we will define formally in the next section, the regulations impact the steady states of the metabolic network.

An example of a regulated metabolic network is shown in Fig. 1. This example is based on a highly simplified model of core carbon metabolism, originally proposed in [9]. At the metabolic level (Fig. 1a), there are 9 metabolites and $m = 9$ reactions. The internal metabolites are Int = {A, D, E, O2, ATP, NADH}, the external metabolites are Ext = {Carbon1, Carbon2, Oxygen}. All the $k = 3$ external metabolites are input metabolites, Ext = Inp. The set of irreversible reactions is $\mathcal{R}$ = {Tc1, Tc2, To2, Td, Te, Growth, Rres, R6, R7}. The stoichiometric coefficients are also given in Fig. 1a. By default, they are set to 1, except for the reactions $R6$ and $R7$.

The regulatory level (Fig. 1b) of the regulated metabolism introduces $d = 2$ regulatory proteins: $\mathcal{P}$ = {RPcl, RPO2}. Thus, the Boolean network f is of dimension $n = k+m+d = 14$. It consists of 14 functions (see Fig. 1b) which map a Boolean vector $x = (x_{\mathrm{Carbon1}}, x_{\mathrm{Carbon2}}, x_{\mathrm{Oxygen}}, x_{\mathrm{RPcl}}, x_{\mathrm{RPO2}}, x_{\mathrm{Tc1}}, x_{\mathrm{Tc2}}, x_{\mathrm{To2}}, x_{\mathrm{Td}}, x_{\mathrm{Te}}, x_{\mathrm{Growth}}, x_{\mathrm{Rres}}, x_{\mathrm{R6}}, x_{\mathrm{R7}}) \in \mathbb{B}^n$ to a Boolean value in $\mathbb{B}$. The local functions associated with regulatory proteins in $\mathcal{P}$ involve only external metabolite variables. Among the 9 functions associated with reactions, only two (Tc2, Rres) are non-constant: they involve the two regulatory proteins.

The influence graph of the network is shown in Fig. 1c. Only the shown nodes (RPcl, RPO2, Tc2, Rres) have a non-constant local function or are used in the local function of another node (Carbon1, Oxygen). The influence graph



(a) *Metabolic Network*

(c) Influence graph $G(f)$ of the regulatory Boolean network f. Nodes without in-going or out-going edges are not represented. Positive edges are drawn in green with a regular tipping arrow, negative edges are drawn in red with a bar arrow.

	Regulatory proteins		Input metabolites		
Local function	$f_{\mathrm{RPO2}}(x)$	$f_{\mathrm{RPcl}}(x)$	$f_{\mathrm{Carbon1}}(x)$	$f_{\mathrm{Carbon2}}(x)$	$f_{\mathrm{Oxygen}}(x)$
Boolean rule	$\neg x_{\mathrm{Oxygen}}$	x_{Carbon1}	0	0	0

	Reactions								
Local function	$f_{\mathrm{Tc1}}(x)$	$f_{\mathrm{Tc2}}(x)$	$f_{\mathrm{To2}}(x)$	$f_{\mathrm{Td}}(x)$	$f_{\mathrm{Te}}(x)$	$f_{\mathrm{Growth}}(x)$	$f_{\mathrm{Rres}}(x)$	$f_{\mathrm{R6}}(x)$	$f_{\mathrm{R7}}(x)$
Boolean rule	1	$\neg x_{\mathrm{RPcl}}$	1	1	1	1	$\neg x_{\mathrm{RPO2}}$	1	1

(b) *Boolean Network*. All Boolean functions equal to 1 correspond to reactions which are not regulated by the Boolean network.

Fig. 1: **Example of regulated metabolic network**. In the metabolic network **(a)**, each node represents a metabolite, and each hyperedge a reaction. For instance, the hyperedge R7 linking {A; NADH} to {E} models the reaction A + 3 NADH → 3 E. Integer values over hyperedges are stoichiometric coefficients, the default value is 1. **(b)** defines the Boolean network regulating the metabolic network in (a), with $x \in \mathbb{B}^n$ and $n = 14$. **(c)** shows the influence (or regulatory) graph of the Boolean network in (b), with square nodes denoting the regulatory proteins.

shows the multi-layered regulations of the network: external input metabolites (Carbon1, Oxygen) regulate regulatory proteins (RPcl, RPO2), which regulate reactions (Tc2, Rres).

2.2 Dynamic rFBA

Flux Balance Analysis (FBA) [19] returns an *optimal* metabolic steady state, according to a given linear objective function in the reaction fluxes. In the following, we assume that the objective function is to maximize the flux through a reaction *Growth*. For regulated metabolic networks, the rFBA framework [9] allows defining a discrete time series of optimal steady states, where regulatory variables can force reaction fluxes to be zero and input metabolite concentrations define upper bounds on uptake fluxes.

Definition 5. *Let $(\mathcal{N}, \mathcal{P}, f)$ be a regulated metabolic network with flux bounds $l_r, u_r \in \mathbb{R}, 0 \leq l_r \leq u_r$, for $r \in \mathcal{R}$. A* metabolic-regulatory steady state *is a triple $(v, w, x) \in \mathbb{R}^{|\mathcal{R}|} \times \mathbb{R}^{|\mathrm{Ext}|} \times \mathbb{B}^{|\mathrm{Inp}|+|\mathcal{R}|+|\mathcal{P}|}$ such that*

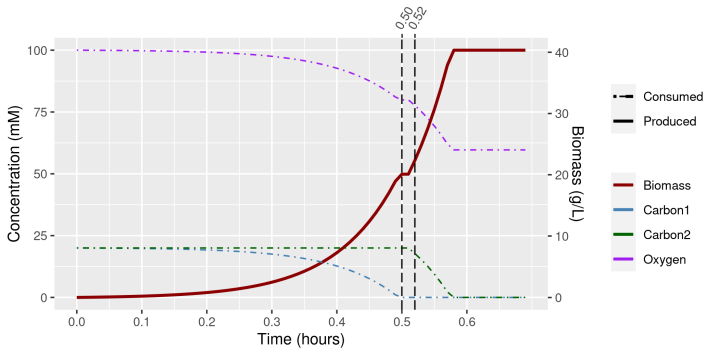
- $S_{\mathrm{Int},\mathcal{R}} \cdot v = 0$,
- *for each reaction $r \in \mathcal{R}$, $l_r \cdot x_r \leq v_r \leq u_r \cdot x_r$,*
- *for each input metabolite $m \in \mathrm{Inp}$ and each reaction $r \in \mathcal{R}$ with $S_{mr} < 0$, $v_r \leq$ uptake_bound(w_m), where uptake_bound(w_m) denotes the maximum flux through uptake reaction r, given the input metabolite concentration w_m.*

Two successive metabolic-regulatory steady states (v^k, w^k, x^k) at time t^k, and $(v^{k+1}, w^{k+1}, x^{k+1})$ at time t^{k+1}, are linked by the following relations:

1. The external metabolite concentrations w^{k+1} are obtained from the previous concentrations w^k by assuming the constant uptake/secretion fluxes v^k for the whole time period $[t^k, t^{k+1}]$.
2. The Boolean state x^{k+1} is obtained by applying the regulatory function f to the binarized input metabolites concentrations $x'_{\mathrm{Inp}} = \beta(w^{k+1}_{\mathrm{Inp}})$ at time t^{k+1}, together with the binarized reaction fluxes $x'_{\mathcal{R}} = \beta(v^k)$ and the Boolean values $x'_{\mathcal{P}} = x^k_{\mathcal{P}}$ of the regulatory proteins at time t^k, *i.e.*,
$$x^{k+1} = f(x')$$
3. $(v^{k+1}, w^{k+1}, x^{k+1})$ is a metabolic-regulatory steady state maximizing the flux through the *Growth* reaction, i.e., there is no metabolic-regulatory steady state (v', w^{k+1}, x^{k+1}) such that $v'_{Growth} > v^{k+1}_{Growth}$.

In this paper, we rely on the FlexFlux implementation of rFBA [18], which assumes a fixed time step τ between successive metabolic-regulatory steady states ($t^{k+1} - t^k = \tau$ for any k). The *Growth* reaction is assumed to reflect the growth of the cell. FlexFlux computes the evolution of the total biomass of the cell as $\mathrm{biomass}^{k+1} = \mathrm{biomass}^k \cdot e^{v^k_{Growth} \cdot \tau}$ (from a given initial $\mathrm{biomass}^0$). The maximum uptake fluxes of input metabolites $m \in \mathrm{Inp}$ at step k are defined as
$$uptake\_bound(w_m) = w_m / (\mathrm{biomass}^k \cdot \tau).$$
Finally, the update of the external metabolite concentrations is computed as
$$w^{k+1}_m = w^k_m - (S_{mr} v^k_r / v^k_{Growth}) \cdot (\mathrm{biomass}^k - \mathrm{biomass}^{k+1}),$$
where $r \in \mathcal{R}$ is the uptake/secretion reaction for the external metabolite m ($S_{mr} < 0$ or $S_{mr} > 0$), which is assumed to be unique.

An example of a dynamic rFBA simulation using FlexFlux of the regulated metabolic network of Fig. 1 is shown in Fig. 2. It uses a time step of $0.01h$ and is initialized with 100 mM of Oxygen, 20 mM of Carbon1 and 20 mM of Carbon2. The simulation shown in Fig. 2a is composed of 70 metabolic steady states. By applying the binarization β, these 70 metabolic steady states correspond to 5



(a) Simulation showing the evolution of the concentrations of the external metabolites (Oxygen, Carbon1, Carbon2) and the production of biomass by the Growth reaction.

		External metabolites			Regulatory proteins		Reaction flows								
Time	w_{biomass}	w_{Carbon1}	w_{Carbon2}	w_{Oxygen}	x_{RPO2}	x_{RPcl}	v_{Tc1}	v_{Tc2}	v_{To2}	v_{Td}	v_{Te}	v_{Growth}	v_{Rres}	v_{R6}	v_{R7}
0.49	17.05	2.95	20.0	82.95	0	1	10.5	0.0	10.5	0.0	0.0	10.5	10.5	0.0	0.0
0.50	18.95	1.05	20.0	81.05	0	1	6.15	0.0	6.15	0.0	0.0	6.15	6.15	0.0	0.0
0.51	20.10	0.0	20.0	79.90	0	0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.52	20.10	0.0	20.0	79.90	0	0	0.0	10.5	10.5	0.0	0.0	10.5	10.5	0.0	0.0
0.53	22.35	0.0	17.76	77.65	0	0	0.0	10.5	10.5	0.0	0.0	10.5	10.5	0.0	0.0

(b) Focus on the times from $0.49h$ to $0.53h$ in the simulation, showing the switch from Carbon1 to Carbon2 for biomass production.

Fig. 2: Dynamic rFBA simulation of the regulated metabolic network in Fig. 1. The simulation is done with FlexFlux and is initialized with 100mM of Oxygen, 20 mM of Carbon1, and 20 mM Carbon2. The time step is set to $0.01h$. The flux bounds are $\forall r \in \{\text{Tc1, Tc2}\}$, $(l_r, u_r) = (0, 10.5)$, $\forall r \in \{\text{Td, Te}\}$, $(l_r, u_r) = (0, 12.0)$, $\forall r \in \{\text{R6, R7, Rres, Growth}\}$, $(l_r, u_r) = (0, 9999)$ and for Oxygen, $(l_r, u_r) = (0, 15.0)$.

different binarized metabolic steady states, which are shown in Tab. 1. These binarized metabolic steady states capture the main features of the simulation.

More precisely, the simulation shows that until $0.5h$ only Carbon1 and Oxygen are consumed to produce biomass. This corresponds to a first time period where the behavior of the system is monotone: the binarized metabolic steady states are equal in this time range. The presence of Carbon1 activates the regulatory protein RPcl inhibiting the reaction Tc2 according to the regulatory rules. At $0.5h$, Carbon1 is depleted and the current Boolean state $x \in \mathbb{B}^{15}$ is such that $x_{\text{Carbon1}} = 0$, $x_{\text{RPcl}} = 1$, $x_{\text{Tc2}} = 0$ (second qualitative behavior with equal binarization of the metabolic steady states). At $0.51h$, as shown in Fig. 2b, the Boolean state x is updated to x' so that the Boolean state of RPcl becomes $x'_{\text{RPcl}} = f_{\text{RPcl}}(x) = x_{\text{Carbon1}} = 0$. The Boolean state of *Tc2* remains unchanged because $x_{\text{RPcl}} = 1$. No biomass is produced at $0.51h$. This corresponds to a third qualitative behavior. At $0.52h$, the Boolean state x' is updated to x'': all the node states remain unchanged except for $x''_{\text{Tc2}} = f_{\text{Tc2}}(x') = \neg x'_{\text{RPcl}} = 1$.

	External metabolites				Regulatory proteins		Reactions								
Time	$\overline{w}_{\text{Biomass}}$	$\overline{w}_{\text{Carbon1}}$	$\overline{w}_{\text{Carbon2}}$	$\overline{w}_{\text{Oxygen}}$	$\overline{x}_{\text{RPO2}}$	$\overline{x}_{\text{RPcl}}$	$\overline{v}_{\text{Tc1}}$	$\overline{v}_{\text{Tc2}}$	$\overline{v}_{\text{To2}}$	$\overline{v}_{\text{Td}}$	$\overline{v}_{\text{Te}}$	$\overline{v}_{\text{Growth}}$	$\overline{v}_{\text{Rres}}$	$\overline{v}_{\text{R6}}$	$\overline{v}_{\text{R7}}$
0	0	1	1	1	0	1	0	0	0	0	0	0	0	0	0
0.01	1	1	1	1	0	1	1	0	1	0	0	1	1	0	0
0.51	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0
0.52	1	0	1	1	0	0	0	1	1	0	0	1	1	0	0
0.59	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0

Table 1: Binarization of the metabolic steady states of simulation in Fig. 2. It contains the binarized values of the metabolic steady state computed by the rFBA simulation. A timepoint t appears in the table if and only if the binarization of the simulated steady state is different from the binarized metabolic steady state of time $t-1$.

This corresponds to a fourth qualitative behavior. The reaction *Tc2* is not inhibited anymore, and biomass is produced due to the uptake of Carbon2 and Oxygen (through *Tc2*, *Growth* and *Rres*) until Carbon2 depletion at $t = 0.59h$ (fifth qualitative behavior).

3 Boolean abstraction of dynamic rFBA

In the previous example, we illustrated how the simulation of a regulated metabolic network may generate time-periods for which the qualitative behavior is similar, meaning that the variation of all the metabolic variables is monotone and the Boolean values of the regulatory proteins are constant. In this section, we introduce a discrete definition of steady states to capture the monotone behaviors observed in rFBA simulations. This allows introducing a discretized form of rFBA, which will be used in the next section for the reverse-engineering framework.

3.1 Boolean metabolic steady states

Given a metabolic network $\mathcal{N} = (\text{Int}, \text{Ext}, \mathcal{R}, S)$, we derive a logical characterization of the notion of steady state, considering that reactions are either inactive or active, and metabolites either absent or present. This will result in a set of *Boolean* metabolic steady states that form an over-approximation of the continuous steady states.

We associate all reactions with propositional variables $\mathcal{V} = \{\overline{v_r}\}_{r\in\mathcal{R}}$. For each metabolite $m \in \text{Int} \uplus \text{Ext}$, we introduce a variable $\overline{z_m}^+$ as a Boolean abstraction of the production of m and a variable $\overline{z_m}^-$ as a Boolean abstraction of the consumption of m:

$$\forall m \in \text{Int} \uplus \text{Ext}, \quad \overline{z_m}^+ \stackrel{\text{def}}{=} \bigvee_{\substack{r\in\mathcal{R},\\ S_{mr}>0}} \overline{v_r}, \qquad \overline{z_m}^- \stackrel{\text{def}}{=} \bigvee_{\substack{r\in\mathcal{R},\\ S_{mr}<0}} \overline{v_r},$$

(where an empty disjunction is considered to be false).

For each internal metabolite m, we introduce a variable $\widehat{z_m}$ which is equal to 1 iff m is in a logical steady state:

$$\forall m \in \mathrm{Int}, \quad \widehat{z_m} \stackrel{\mathrm{def}}{=} (\overline{z_m}^{+} \Leftrightarrow \overline{z_m}^{-}).$$

For the external metabolites, we introduce propositional variables $\mathcal{V}_{ext} = \{\overline{z_m}\}_{m \in \mathrm{Ext}}$ indicating whether or not m is present in the environment. The formula

$$\widehat{\mathcal{N}}_{\mathrm{Ext}} \stackrel{\mathrm{def}}{=} \bigwedge_{m \in \mathrm{Ext}} (\overline{z_m}^{-} \Rightarrow \overline{z_m})$$

then states that an external metabolite can only be consumed if it is present in the environment.

Definition 6 (Boolean metabolic steady state). *A* Boolean metabolic steady state *of a metabolic network* $\mathcal{N} = (\mathrm{Int}, \mathrm{Ext}, \mathcal{R}, S)$ *is a Boolean vector* $\hat{\nu} \in \mathbb{B}^{|\mathrm{Ext}|+|\mathcal{R}|}$ *which is a satisfying assignment of the following logical steady state formula:*

$$\widehat{\mathcal{N}} \stackrel{def}{=} \widehat{\mathcal{N}}_{\mathrm{Ext}} \wedge \bigwedge_{m \in \mathrm{Int}} \widehat{z_m}$$

We denote by $\mathrm{MSS}^{\mathbb{B}}(\mathcal{N}) \subseteq \mathbb{B}^{|\mathrm{Ext}|+|\mathcal{R}|}$ *the set of all the Boolean metabolic steady states of the metabolic network* $\mathcal{N}$*.*

As an immediate consequence of this definition, we get the following property:

Property 1. For each metabolic-regulatory steady state (v, w, x) of the regulated metabolic network $(\mathcal{N}, \mathcal{P}, f)$, the binarized value $\beta(w, v)$ of the external metabolite concentrations w and the reaction fluxes v is a Boolean metabolic steady state, i.e., $\beta(w, v) \in \mathrm{MSS}^{\mathbb{B}}(\mathcal{N})$.

Note that the converse is not true: since the logical characterization neglects the stoichiometry, Boolean metabolic steady states may have no real-valued counterpart.

Applied to the example, the internal metabolic constraints are the following:

$\overline{z_{\mathrm{A}}}^{+} = \overline{v_{\mathrm{Tc1}}} \vee \overline{v_{\mathrm{Tc2}}}, \quad \overline{z_{\mathrm{A}}}^{-} = \overline{v_{\mathrm{R6}}} \vee \overline{v_{\mathrm{R7}}} \vee \overline{v_{\mathrm{Growth}}}$
$\overline{z_{\mathrm{D}}}^{+} = \overline{v_{\mathrm{R6}}}, \quad \overline{z_{\mathrm{D}}}^{-} = \overline{v_{\mathrm{Td}}}, \quad \overline{z_{\mathrm{E}}}^{+} = \overline{v_{\mathrm{R7}}}, \quad \overline{z_{\mathrm{E}}}^{-} = \overline{v_{\mathrm{Te}}}$
$\overline{z_{\mathrm{O2}}}^{+} = \overline{v_{\mathrm{To2}}}, \quad \overline{z_{\mathrm{O2}}}^{-} = \overline{v_{\mathrm{Rres}}}$
$\overline{z_{\mathrm{ATP}}}^{+} = \overline{v_{\mathrm{R6}}} \vee \overline{v_{\mathrm{Rres}}}, \quad \overline{z_{\mathrm{ATP}}}^{-} = \overline{v_{\mathrm{Growth}}}$
$\overline{z_{\mathrm{NADH}}}^{+} = \overline{v_{\mathrm{Growth}}}, \quad \overline{z_{\mathrm{NADH}}}^{-} = \overline{v_{\mathrm{R7}}} \vee \overline{v_{\mathrm{Rres}}}$

The logical steady state constraints equivalent to $\widehat{\mathcal{N}} = 1$ are obtained by gathering contraints on internal and external metabolites:

$\overline{v_{\mathrm{Tc1}}} \vee \overline{v_{\mathrm{Tc2}}} = \overline{v_{\mathrm{R6}}} \vee \overline{v_{\mathrm{R7}}} \vee \overline{v_{\mathrm{Growth}}}$
$\overline{v_{\mathrm{R6}}} = \overline{v_{\mathrm{Td}}} \qquad \overline{v_{\mathrm{R7}}} = \overline{v_{\mathrm{Te}}} \qquad \overline{v_{\mathrm{To2}}} = \overline{v_{\mathrm{Rres}}}$
$\overline{v_{\mathrm{R6}}} \vee \overline{v_{\mathrm{Rres}}} = \overline{v_{\mathrm{Growth}}} \qquad \overline{v_{\mathrm{R7}}} \vee \overline{v_{\mathrm{Rres}}} = \overline{v_{\mathrm{Growth}}}$
$\overline{v_{\mathrm{Tc1}}} \Rightarrow \overline{z_{\mathrm{Carbon1}}} \qquad \overline{v_{\mathrm{Tc2}}} \Rightarrow \overline{z_{\mathrm{Carbon2}}} \qquad \overline{v_{\mathrm{To2}}} \Rightarrow \overline{z_{\mathrm{Oxygen}}}$

From these equations, we deduce that there are 38 Boolean metabolic steady states for the example shown in Fig. 1. These Boolean metabolic steady states are detailed in Appendix A. Among them, we recover the five binarized metabolic-regulatory steady states (Table 1) appearing in the rFBA simulations of Fig.2.

3.2 Boolean dynamics

Using the logical characterization of metabolic steady states, we define a Boolean counterpart of dynamic rFBA (Sect. 2.2). A Boolean state of the regulated metabolic network $(\mathcal{N}, \mathcal{P}, f)$ assigns a Boolean value to external metabolites, reactions, and regulatory proteins, which gives a Boolean vector of dimension $n = k + m + d$. Such a Boolean state $x \in \mathbb{B}^n$ should match with a Boolean metabolic steady state. Denoting by $\mathcal{M} = \text{Ext} \cup \mathcal{R}$ the external metabolites and reactions, $x_{\mathcal{M}}$ should verify the Boolean metabolic steady state constraints described in the previous section ($x_{\mathcal{M}} \in \text{MSS}^{\mathbb{B}}(\mathcal{N})$). The general idea is then to capture the possible successions of such Boolean states, subject to the regulations through the regulatory proteins specified by the Boolean network f.

A key ingredient of dynamic rFBA is the objective function to maximize, typically the fluxes of reactions producing biomass. However, at the Boolean level, it is not possible to directly rank metabolic steady states according to their biomass production, as this will be either absent or present. Thus, a specific *Boolean objective function* has to be provided to score a Boolean metabolic steady state. This takes the form of a function $\hat{o}$ mapping Boolean metabolic steady states to natural numbers: $\hat{o} : \mathbb{B}^{k+m} \to \mathbb{N}$. The Boolean dynamics will only select Boolean metabolic steady states maximizing this supplied objective.

When considering possible next states, it is crucial to account for those where the input metabolites change their value. Hereafter, we consider any possible change.

The Boolean dynamic rFBA is formalized by a function $\text{next}^{\mathbb{B}}_{(\mathcal{N},\mathcal{P},f,\hat{o})}$ which associates any Boolean state of the regulated metabolic network to a set of admissible next states:

Definition 7 (Boolean dynamic rFBA: $\text{next}^{\mathbb{B}}_{(\mathcal{N},\mathcal{P},f,\hat{o})} : \mathbb{B}^n \to 2^{\mathbb{B}^n}$). *For any Boolean states $x, y \in \mathbb{B}^n$, $y \in \text{next}^{\mathbb{B}}_{(\mathcal{N},\mathcal{P},f,\hat{o})}(x)$ if and only if for $x' = (y_{\text{Inp}}, x_{\mathcal{R}\cup\mathcal{P}}) \in \mathbb{B}^n$,*

1. *the values of the regulatory proteins are computed synchronously from x' according to f: $y_{\mathcal{P}} = f_{\mathcal{P}}(x')$,*
2. *y matches with a Boolean metabolic steady state: $y_{\mathcal{M}} \in Z(x')$, and*
3. *the matching Boolean metabolic steady state maximizes the supplied objective function: $\forall y'_{\mathcal{M}} \in Z(x'), \hat{o}(y_{\mathcal{M}}) \geq \hat{o}(y'_{\mathcal{M}})$.*

Here $Z(x') = \{z \in \text{MSS}^{\mathbb{B}}(\mathcal{N}) \mid z_{\text{Inp}} = x'_{\text{Inp}}, z_{\mathcal{R}} \preceq f_{\mathcal{R}}(x')\}$ is the set of Boolean metabolic steady states that match with the value of external metabolites and with the regulations from x'.

Let us consider the regulated metabolic network from Fig. 1. It appears that the steady states maximizing the growth maximize the input fluxes. Thus, we set the Boolean objective function $\hat{o}$ as the sum of input reactions:

$$\hat{o}(x) = x_{\mathrm{Tc1}} + x_{\mathrm{Tc2}} + x_{\mathrm{To2}} \ .$$

Consider the Boolean state from Table 1 at time 0, which we name x, and the next Boolean state at time 0.51, which we name y, with the same input metabolite values ($x_{\mathrm{Inp}} = y_{\mathrm{Inp}}$). Using the notation from the above definition, we set $x' = x$. Imagine the case where no reactions is regulated, i.e., the regulatory BN is of the form $f'_r(x) = 1$ for every $r \in \mathcal{R}$. Among the Boolean metabolic steady states z matching the input values ($z_{\mathrm{Inp}} = x'_{\mathrm{Inp}}$), the ones that maximize $\hat{o}$ always verify $z_{\mathrm{Tc2}} = 1$ (Boolean metabolic steady states 26, 29, 32, 38 in the Table 3 in Appendix A), which does not match with y. Thus y would not be an admissible next state.

Considering now the regulatory BN f of Fig. 1, we obtain $f_{\mathrm{Tc2}}(x') = \neg x'_{\mathrm{RPcl}} = 0$ and for each other reaction $r \in \mathcal{R} \setminus \{\mathrm{Tc2}\}$, $f_r(x') = 1$. The set $Z(x')$ contains 4 matching optimal Boolean steady states (rows 25, 28, 31, 37 of Table A.3), among them the one matching with y. Thus $y \in \mathrm{next}^{\mathbb{B}}_{(\mathcal{N},\mathcal{P},f,\hat{o})}(x)$.

Let x be now the Boolean state at time 0.01, and y the next Boolean state at time 0.51, where the input metabolites have a different state (Carbon1 switched to 0). Let x' be equal to x except for the input metabolites, which are equal to y_{Inp}. We obtain that $f_{\mathrm{RPO2,RPcl}}(x') = (\neg x'_{\mathrm{Oxygen}}, x'_{\mathrm{Carbon1}}) = (0,0) = y_{\mathrm{RPO2,RPcl}}$. Moreover, $f_{\mathrm{Tc2}}(x') = \neg x'_{\mathrm{RPcl}} = 0$ and for each other reaction $r \in \mathcal{R}$, $r \neq \mathrm{Tc2}$, $f_r(x') = 1$. In this case, there is only one Boolean metabolic steady state z such that $z_{\mathrm{Inp}} = x'_{\mathrm{Inp}}$ and $z_{\mathcal{R}} \preceq f_{\mathcal{R}}(x')$. It appears that it matches with y, i.e., $z = y_{\mathcal{M}}$; thus $y \in \mathrm{next}^{\mathbb{B}}_{(\mathcal{N},\mathcal{P},f,\hat{o})}(x)$.

4 Inference of regulations from rFBA time series

Given sequences of metabolic-regulatory steady states obtained by dynamic rFBA from a ground-truth regulated metabolic network under different conditions, our objective is to infer all the regulatory Boolean networks that can reproduce the observed behaviors. Besides the ground-truth model, the inference may suggest alternative regulatory logics.

Definition 8 (Search domain for BNs). *The search domain for BNs, denoted by $\mathbb{F}$, is constrained by an influence graph $\mathcal{G}$: any candidate $f \in \mathbb{F}$ should satisfy $G(f) \subseteq \mathcal{G}$,* i.e. *uses at most the influences allowed in $\mathcal{G}$. Moreover, we assume that f is locally monotone.*

Typically, $\mathcal{G}$ contains the putative influences from and to regulatory proteins. In our case study, $\mathcal{G}$ is obtained from the ground-truth regulatory model f° by "forgetting" the sign of influences (for each $(i,s,j) \in G(f^\circ)$, $\{(i,+,j),(i,-,j)\} \subseteq \mathcal{G}$), and adding putative influences.

Our inference problem mixes both linear constraints for characterizing the optimal steady states of the metabolic network with Boolean constraints for

characterizing the value changes of regulatory proteins. To express the inference problem, we rely on the Boolean abstraction of dynamic rFBA presented in the previous section.

4.1 Approximation as a Boolean satisfiability problem

We propose a relaxation of the inference problem by the means of the Boolean dynamic rFBA interpretation given in Sect. 3.

Inputs of the relaxed inference problem. The inputs of the problem are **(i)** a metabolic network $\mathcal{N}$ and a set of regulatory proteins $\mathcal{P}$, **(ii)** sequences of metabolic-regulatory steady states, represented by sets of pairs (s^t, s^{t+1}), with $s^t = (v^t, w^t, x^t)$ and $s^{t+1} = (v^{t+1}, w^{t+1}, x^{t+1})$ following the notation from Def. 5: the observed changes of metabolic-regulatory steady states are given as $T \subseteq \mathbb{S}\times\mathbb{S}$ with $\mathbb{S} = \mathbb{R}^{|\text{Inp}|+|\mathcal{R}|} \times \mathbb{B}^{|RPs|}$, **(iii)** a domain of putative regulatory BNs $\mathbb{F}$ of dimension $n = |\text{Inp}| + |\mathcal{R}| + |\mathcal{P}|$, **(iv)** a Boolean state objective score $\hat{o} : \mathbb{B}^n \to \mathbb{N}$.

Relaxed inference problem The relaxed inference problem consists then in identifying the $f \in \mathbb{F}$ such that for each $(s, s') \in T$,

$$\beta(s') \in \text{next}^{\mathbb{B}}_{(\mathcal{N},\mathcal{P},f,\hat{o})}(\beta(s)).$$

Formulation as a satisfiability problem. Relying on the Boolean dynamic rFBA abstraction, the inference problem boils down to a satisfiability problem in propositional Boolean logic using two levels of quantifiers (2-QBF):

$$\exists f \in \mathbb{F}, \forall (s, s') \in T, \exists y \in \text{MSS}^{\mathbb{B}}(\mathcal{N}),\ y_{\text{Inp}} = x'_{\text{Inp}}, y_{\mathcal{P}} = f_{\mathcal{P}}(x'),\ y_{\mathcal{R}} \preceq f_{\mathcal{R}}(x'),$$
$$\forall z \in \text{MSS}^{\mathbb{B}}(\mathcal{N}), (z_{\text{Inp}} \neq x'_{\text{Inp}} \vee z_{\mathcal{P}} \neq f_{\mathcal{P}}(x') \vee z_{\mathcal{R}} \not\preceq f_{\mathcal{R}}(x') \vee \hat{o}(z) \leq \hat{o}(y))$$

with $x' \in \mathbb{B}^n$ defined as $x'_{\text{Inp}} = \beta(s')_{\text{Inp}}$ and $x'_{\mathcal{R}\cup\mathcal{P}} = \beta(s)_{\mathcal{R}\cup\mathcal{P}}$.

Note that without the Boolean optimization criteria $\hat{o}$ (equivalently $\hat{o}(z) = c$), the problem reduces to a SAT problem where the only constraints relate to the local functions of the regulatory proteins:

$$\exists f \in \mathbb{F}, \exists y \in \text{MSS}^{\mathbb{B}}(\mathcal{N}),\ y_{\text{Inp}} = x'_{\text{Inp}},\ y_{\mathcal{P}} = f_{\mathcal{P}}(x')$$

Indeed, $y_{\mathcal{R}} \preceq f_{\mathcal{R}}(x')$ is always verified whenever $f_r(x) = 1$ for each $r \in \mathcal{R}$.

Since the Boolean dynamic rFBA gives an over-approximation of metabolic steady states, and even assuming that the Boolean objective function $\hat{o}$ matches with the optimal metabolic steady states, our formulation leads to an approximation of admissible regulatory BN f: it may happen that a spurious Boolean metabolic steady state (having no real counter part) has a strictly higher value with $\hat{o}$ than non-spurious ones.

4.2 Implementation in Answer-Set Programming

Answer-Set Programming (ASP) [1,12] is a declarative framework allowing solving combinatorial satisfaction problems. It relies on the stable model semantics [10]. The basic idea of ASP is to express a problem in a logical format so that the (logic) models of its representation provide the solutions to the original problem. Problems are expressed as logic programs (first order logic predicates expressed with rules with the shape `<head> :- <body> .`). Stable models of the logic programs are referred to as *answer sets*. Although determining whether a program has an answer set is the fundamental decision problem in ASP, modern ASP solvers like clingo [13] support various combinations of reasoning modes, among them, regular and projective enumeration, intersection and union, multi-criteria optimization and subset minimal and maximal model enumeration [15].

The stable model semantics of ASP combined with disjunctive programming are the key ingredients that enable expressing two quantification levels Boolean formulas (2-QBF problem), *i.e.* $\exists x$, $\forall y$, $\phi(x, y)$ where $\phi(x, y)$ is a quantifier-free propositional formula (Σ_2^{P}-complete) [10]. The encoding of 2-QBF relies on the so-called *saturation technique* [11,14]. Essentially, for fixed x and y, the encoding ensures that a maximal (saturated) answer-set is returned if and only if $\phi(x, y)$. Thus, whenever there exists y such that $\phi(x, y)$ does not hold (counter-example), a smaller answer-set is returned. Following the subset-minimal stable semantics, the 2-QBF problem is satisfiable if and only if only saturated answer-set are subset-minimal.

5 Case study

As a proof of concept, we apply our inference framework to the simplified core carbon metabolism described in Fig. 1. First, from this ground-truth model, we generate sample dynamic rFBA simulations for different input conditions, reproducing existing biological observations [9]. Next we take these simulations as input for our method, together with an influence graph extending the one from the ground truth model with additional putative regulations. Using our inference method, we then enumerate BNs that are compatible with both the simulations and the influence graph. The results show that the ground truth model is well recovered, together with some alternative BNs. In particular, a simpler BN matching the data is identified, which uses fewer regulations. It turns out that the missing regulation is not needed to reproduce the expected biological behavior. Our implementation relying on the ASP solver CLINGO [13] together with the case study is available at `https://github.com/bioasp/boolean-caspo-flux`. They can be reproduced using the notebooks and docker image at `https://doi.org/10.5281/zenodo.5060984`.

Input simulations We designed six dynamic r-FBA simulations of the BN of Fig. 1(b) to mimic the studies of the core carbon metabolism in [9]. They correspond to different sets of initially available input metabolites and regulatory proteins (Table 3a, and Fig. 4 in Appendix B). For instance, Experiment 1

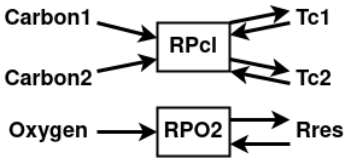
assumes that all input metabolites (Carbon1, Carbon2, Oxygen) are available. Experiment 2 assumes that Carbon1 and Carbon2 are present at initialization but not Oxygen.

For each case, we use FlexFlux with an initial biomass value of 0.1 and a time step of 0.01 to simulate the system. Each of the 6 simulations involves 200 metabolic steady states. For initial external metabolite values ($\overline{z_{\text{Carbon1}}}$, $\overline{z_{\text{Carbon2}}}$, $\overline{z_{\text{Oxygen}}}$), the regulatory proteins are initialized such that $x_{\text{RPcl}} = \overline{z_{\text{Carbon1}}}$ and $x_{\text{RPO2}} = \neg\overline{z_{\text{Oxygen}}}$ (Table 3a). Each simulation $S = \{(v, w, x)_0, ..., (v, w, x)_{200}\}$ includes 201 continuous metabolic-regulatory steady states (1 for the initialization and 200 for the simulation). The simulations are then binarized with $S^{\mathbb{B}} = \{(\overline{v_t}, \overline{z_t}) = \beta((v_t, w_t)) \mid \forall v_t \in S\}$, and consecutive identical Boolean states are removed. Table 1 shows the binarized metabolic-regulatory steady states from the simulation of the first experiment. From the 201 continuous metabolic steady states, 5 Boolean metabolic-regulatory steady states remain, corresponding to the time steps {0, 1, 51, 52, 59} (see Table 4 in Appendix B for the resulting states in each simulation).

Candidate models The search domain $\mathbb{F}$ for the candidate BNs is delimited by the influence graph $\mathcal{G}$ of Fig. 3b, which extends the influence graph from the ground-truth model by additional putative regulations, and by relaxing the sign constraints. Since the influence graph $G(f)$ of the ground-truth BN f is included in $\mathcal{G}$, we have $f \in \mathbb{F}$. In addition, $\mathbb{F}$ contains all the BNs such that $f_i(x) = 1$, for all $i \in \text{Inp} \cup \mathcal{R} \setminus \{\text{Tc1, Tc2, Rres}\}$. Furthermore, f_{RPcl} can depend on Carbon1, Carbon2, Tc1, and Tc2, f_{RPO2} can depend on Oxygen, Rres, f_{Tc1} and f_{Tc2} can depend on RPcl, and f_{Rres} can depend on Rres. Overall, $\mathbb{F}$ contains $\prod_{n \in \text{node}(\mathcal{G})} M(\delta^-(n)) = 1\,944\,320$ BNs, with $\delta^-(n)$ the in-degree of n and $M(i)$ the number of monotone Boolean functions with i inputs (Dedekind number).

	Input Metabolite			Regulatory Protein	
Experiment	$\overline{z_{\text{Carbon1}}}$	$\overline{z_{\text{Carbon2}}}$	$\overline{z_{\text{Oxygen}}}$	x_{RPcl}	x_{RPO2}
1	1	1	1	1	0
2	1	1	0	1	1
3	0	1	0	0	1
4	1	0	0	1	1
5	1	0	1	1	0
6	0	1	1	0	0

(a) Initial states of the six rFBA simulations used to create the dataset for the case study.



(b) Influence graph $\mathcal{G}$ delimiting the domain of putative regulatory BNs $\mathbb{F}$. Nodes without in-going or out-going edges are not represented. Black regular tipping arrows are unsigned edges, *i.e.* both positive and negative edges.

Fig. 3: Input data for the case study. Table (a) summarizes the experimental conditions used to generate the input simulations. Figure (b) shows the influence graph delimiting the search domain for the inference problem.

	$f_{\mathrm{RPO2}}(x)$	$f_{\mathrm{RPcl}}(x)$	$f_{\mathrm{Tc1}}(x)$	$f_{\mathrm{Tc2}}(x)$	$f_{\mathrm{Rres}}(x)$	Subset minimal	Ground truth
Model 1	$\neg x_{\mathrm{Oxygen}}$	x_{Carbon1}	1	$\neg x_{\mathrm{RPcl}}$	1	✓	
Model 2	$\neg x_{\mathrm{Oxygen}}$	x_{Carbon1}	1	$\neg x_{\mathrm{RPcl}}$	$\neg x_{\mathrm{RPO2}}$		✓
Model 3	$\neg x_{\mathrm{Oxygen}}$	x_{Carbon1}	x_{RPcl}	$\neg x_{\mathrm{RPcl}}$	1		
Model 4	$\neg x_{\mathrm{Oxygen}}$	x_{Carbon1}	x_{RPcl}	$\neg x_{\mathrm{RPcl}}$	$\neg x_{\mathrm{RPO2}}$		

Table 2: Inferred models having subset minimal local functions. The not shown local functions $f_{\mathrm{Carbon1}}(x)$, $f_{\mathrm{Carbon2}}(x)$, $f_{\mathrm{Oxygen}}(x)$, $f_{\mathrm{To2}}(x)$, $f_{\mathrm{Td}}(x)$, $f_{\mathrm{Te}}(x)$, $f_{\mathrm{Growth}}(x)$, $f_{\mathrm{R6}}(x)$, $f_{\mathrm{R7}}(x)$ are set to 1.

Boolean objective function Our inference framework requires defining an objective function $\hat{o}$ over the Boolean metabolic steady states. Given the set of input metabolites Inp = {Carbon1, Carbon2, Oxygen}, the objective function is defined as $\hat{o}(x) = \sum_{e \in \mathrm{Inp}} x_{\mathrm{e}}, \forall x \in \mathrm{MSS}^{\mathbb{B}}(\mathcal{N})$. This is motivated by the observation that maximizing biomass production often corresponds to maximizing the uptake of inputs according to the QSS constraints. Therefore, if an available input metabolite is not used in the observed Boolean metabolic network, then this must be explained by at least one regulation. This objective function allows capturing more refined behaviors at the discrete level than a standard biomass optimization function, which may be too rough when considering discretized values.

Results Applying the constraints from above allows inferring 40 models. All these models share 3 local functions whose value is not constantly 1 ($f_{\mathrm{RPO2}}(x)$, $f_{\mathrm{RPcl}}(x)$, $f_{\mathrm{Tc2}}(x)$). They also share 9 local functions equal to 1 ($f_{\mathrm{Carbon1}}(x)$, $f_{\mathrm{Carbon2}}(x)$, $f_{\mathrm{Oxygen}}(x)$, $f_{\mathrm{To2}}(x)$, $f_{\mathrm{Td}}(x)$, $f_{\mathrm{Te}}(x)$, $f_{\mathrm{Growth}}(x)$, $f_{\mathrm{R6}}(x)$, $f_{\mathrm{R7}}(x)$). Finally, 2 functions can be set both to 1 or different from 1 according to the model. The 4 *smallest* inferred models are described in Table 2. They can be considered as the smallest because each local function f_i of these 4 models is contained in the local function f_i of the 36 other models. Note that the ground truth, *i.e.* the model used to generate the input data, is correctly inferred (Model 2).

As we represent the local Boolean functions using their disjunctive normal form (DNF), we can focus on the *simplest* models by looking at the *subset-minimal* ones: a Boolean function f_i is smaller than a Boolean function g_i if each of the clauses of f_i is a subset of a clause of g_i. In this case study, there is a single subset-minimal model: the BN 1 of Table 2. The two functions $f_{\mathrm{Rres}}(x)$, $f_{\mathrm{Tc1}}(x)$ are set to 1 due to the subset-minimal constraint. The inferred model is thus $f_{\mathrm{RPO2}}(x) = \neg x_{\mathrm{Oxygen}}$, $f_{\mathrm{RPcl}}(x) = x_{\mathrm{Carbon1}}$, $f_{\mathrm{Tc2}}(x) = \neg x_{\mathrm{RPcl}}$ and all the others local functions are set to 1. Note that only $f_{\mathrm{Rres}}(x)$ differs between the inferred subset-minimal model and the ground truth model.

In order to check whether this subset-minimal model could be considered as an alternative to the ground truth one, we performed dynamic rFBA simulations with the six experimental conditions described in Table 3a. We observe that the resulting time series are strictly identical to the simulations of the ground truth model used to generate the dataset. This suggests that the regulation on *Rres* is not necessary to reproduce the observed behaviors. The proposed subset-minimal

model allows inferring all the needed regulations and can be considered as the simplest regulated metabolic model matching the experimental conditions of Table 3a. Already in [9], the authors recognize that unlike others regulations, *Rres* "regulation is not necessary for the solution". Biologically, this regulation is only present to ensure that unnecessary enzymes decay. However, since enzyme amounts are not explicitly represented in the rFBA framework, the dataset does not reflect this biologic behavior, making it impossible to infer properly the regulation. Taking into account enzymatic resources using methods such as r-deFBA [17], should allow solving this issue. However, the inference approach will also have to be adapted to this kind of extended metabolic modeling.

6 Discussion

We proposed a formal framework to infer Boolean rules for the regulation of a metabolic network. The formulation of dynamic rFBA as sequence of steady states of the regulated metabolic network enables inferring the Boolean rules from time series under multiple conditions. A proof of concept was performed on the simulation of the diauxic shift in carbon metabolism on a small model.

Our method builds on a Boolean abstraction of the dynamic rFBA framework. It enables a formulation of the inference problem as a pure Boolean satisfiability problem using two levels of quantifiers, which can be efficiently solved using Answer Set Programming. One important parameter is the Boolean objective function, which aims at identifying Boolean metabolic steady states that match the optimal real-valued ones. This function is currently specified manually, based on biological expertise. Future work may explore how to derive an objective function automatically. An alternative direction is to solve directly the inference problem by mixing linear programming and Boolean constraints. Future work will investigate the scalability of solving these different inference problems.

Several other perspectives are to be explored. First, all regulations were considered as synchronous, which may not be the case *in vivo*, where regulations can have different time scales. This choice was actually imposed by the use of the FlexFlux implementation. Nevertheless, our method can be easily adapted to support fully-asynchronous and asynchronous updating modes, enabling potential alternative solutions. Second, the production and degradation times of regulatory proteins and enzymes were not taken into account. Moreover, the regulations were considered to be binary. However, we know that metabolism proceeds by finer regulations than the abstraction proposed here, as captured for instance by regulatory dynamic enzyme-cost FBA [17].

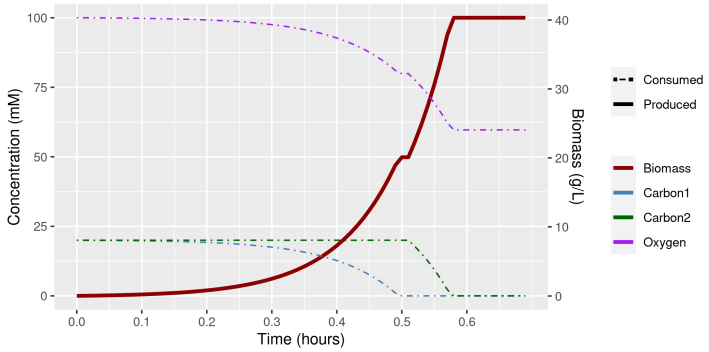
Acknowledgments Work of LC and CB is supported by the French Laboratory of Excellence project "TULIP" (grant number ANR-10-LABX-41; ANR-11-IDEX-0002-02). Work of LP is supported by the French Agence Nationale pour la Recherche (ANR) in the scope of the project "BNeDiction" (grant number ANR-20-CE45-0001).

A Binarized metabolic steady state

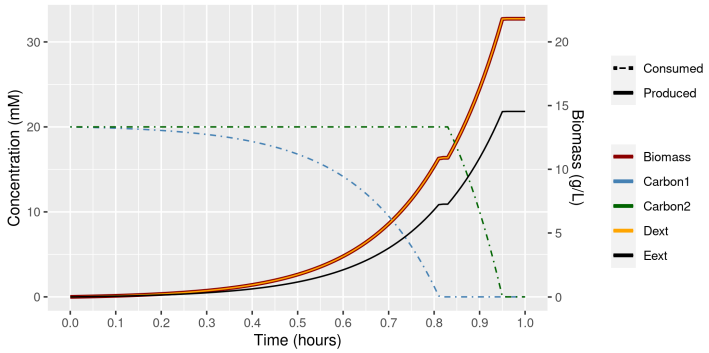
	External metabolites			Reactions									
	$\overline{z_{\text{Carbon1}}}$	$\overline{z_{\text{Carbon2}}}$	$\overline{z_{\text{Oxygen}}}$	$\overline{v_{\text{Tc1}}}$	$\overline{v_{\text{Tc2}}}$	$\overline{v_{\text{To2}}}$	$\overline{v_{\text{Td}}}$	$\overline{v_{\text{Te}}}$	$\overline{v_{\text{Growth}}}$	$\overline{v_{\text{Rres}}}$	$\overline{v_{\text{R6}}}$	$\overline{v_{\text{R7}}}$	**Experimentation**
1	0	0	0	0	0	0	0	0	0	0	0	0	2, 3, 4
2	0	0	1	0	0	0	0	0	0	0	0	0	1, 5, 6
3	0	1	0	0	0	0	0	0	0	0	0	0	2, 3
4	0	1	0	0	1	0	1	1	1	0	1	1	2, 3
5	0	1	1	0	0	0	0	0	0	0	0	0	1, 6
6	0	1	1	0	1	1	0	0	1	1	0	0	1, 6
7	0	1	1	0	1	1	0	1	1	1	0	1	
8	0	1	1	0	1	1	1	0	1	1	1	0	
9	0	1	1	0	1	0	1	1	1	0	1	1	
10	0	1	1	0	1	1	1	1	1	1	1	1	
11	1	0	0	0	0	0	0	0	0	0	0	0	4
12	1	0	0	1	0	0	1	1	1	0	1	1	4
13	1	0	1	0	0	0	0	0	0	0	0	0	5
14	1	0	1	1	0	1	0	0	1	1	0	0	5
15	1	0	1	1	0	1	0	1	1	1	0	1	
16	1	0	1	1	0	1	1	0	1	1	1	0	
17	1	0	1	1	0	0	1	1	1	0	1	1	
18	1	0	1	1	0	1	1	1	1	1	1	1	
19	1	1	0	0	0	0	0	0	0	0	0	0	2
20	1	1	0	0	1	0	1	1	1	0	1	1	
21	1	1	0	1	0	0	1	1	1	0	1	1	2
22	1	1	0	1	1	0	1	1	1	0	1	1	
23	1	1	1	0	0	0	0	0	0	0	0	0	1
24	1	1	1	0	1	1	0	0	1	1	0	0	
25	1	1	1	1	0	1	0	0	1	1	0	0	1
26	1	1	1	1	1	1	0	0	1	1	0	0	
27	1	1	1	0	1	1	0	1	1	1	0	1	
28	1	1	1	1	0	1	0	1	1	1	0	1	
29	1	1	1	1	1	1	0	1	1	1	0	1	
30	1	1	1	0	1	1	1	0	1	1	1	0	
31	1	1	1	1	0	1	1	0	1	1	1	0	
32	1	1	1	1	1	1	1	0	1	1	1	0	
33	1	1	1	0	1	0	1	1	1	0	1	1	
34	1	1	1	1	0	0	1	1	1	0	1	1	
35	1	1	1	1	1	0	1	1	1	0	1	1	
36	1	1	1	0	1	1	1	1	1	1	1	1	
37	1	1	1	1	0	1	1	1	1	1	1	1	
38	1	1	1	1	1	1	1	1	1	1	1	1	

Table 3: All the Boolean metabolic steady states admissible for the metabolic network $\mathcal{N}$ show Fig. 1a. The external metabolite *Biomass* is not shown since its value can be both 0 and 1 for each Boolean metabolic steady state. The experimentation column indicates the numbers of the experiments where the Boolean metabolic steady states occurs.

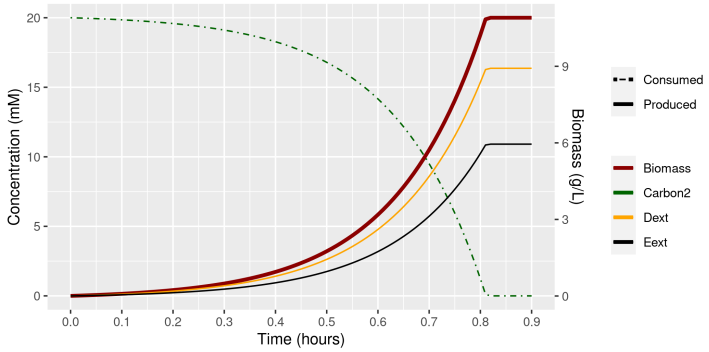
B Experiments and simulations



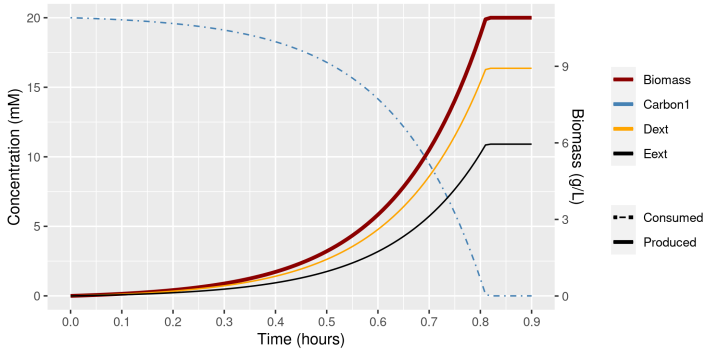
(a) Simulation of experiment 1.



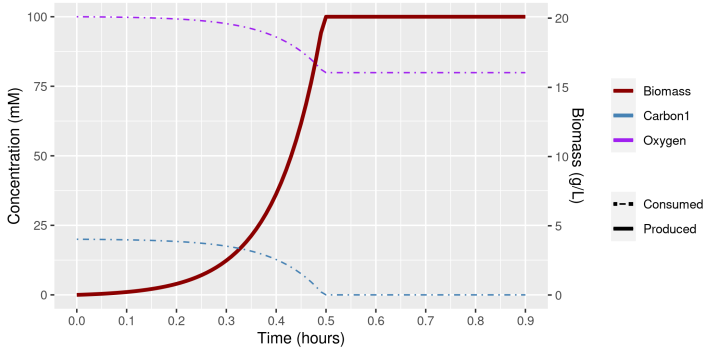
(b) Simulation of experiment 2.



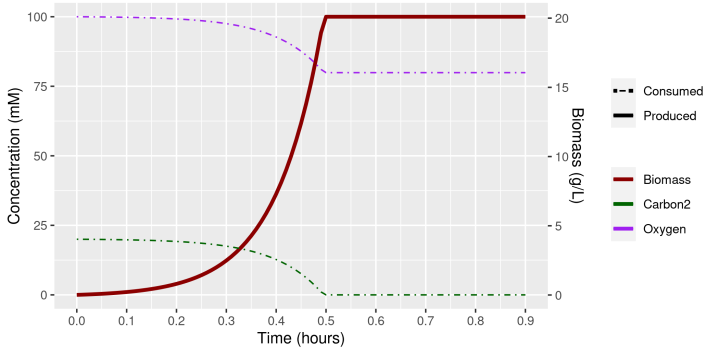
(c) Simulation of experiment 3.



(d) Simulation of experiment 4.



(e) Simulation of experiment 5.



(f) Simulation of experiment 6.

Fig. 4: Simulation made with FlexFlux of the regulated metabolic network in Fig. 1 for each experiment (Table 3a). Time step is set to 0.01. Reaction domains are $\forall r \in \{\text{Tc1, Tc2}\}$, $(l_r, u_r) = (0, 10.5)$, $\forall r \in \{\text{Td, Te}\}$, $(l_r, u_r) = (0, 12.0)$, $\forall r \in \{\text{R6, R7, Rres, Growth}\}$, $(l_r, u_r) = (0, 9999)$ and for Oxygen, $(l_r, u_r) = (0, 15.0)$.
The same simulation graphs are obtained using the local function $f_{\text{Rres}} = \neg x_{\text{RPO2}}$ and $f_{\text{Rres}} = 1$.

		External metabolites				Regulatory proteins		Reactions								
Experiment	Time	$\overline{z_{\text{Biomass}}}$	$\overline{z_{\text{Carbon1}}}$	$\overline{z_{\text{Carbon2}}}$	$\overline{z_{\text{Oxygen}}}$	$\overline{x_{\text{RPO2}}}$	$\overline{x_{\text{RPci}}}$	$\overline{v_{\text{Tc1}}}$	$\overline{v_{\text{Tc2}}}$	$\overline{v_{\text{To2}}}$	$\overline{v_{\text{Td}}}$	$\overline{v_{\text{Te}}}$	$\overline{v_{\text{Growth}}}$	$\overline{v_{\text{Rres}}}$	$\overline{v_{\text{R6}}}$	$\overline{v_{\text{R7}}}$
1	0	0	1	1	1	0	1	0	0	0	0	0	0	0	0	0
	1	1	1	1	1	0	1	1	0	1	0	0	1	1	0	0
	51	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0
	52	1	0	1	1	0	0	0	1	1	0	0	1	1	0	0
	59	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0
2	0	0	1	1	0	1	1	0	0	0	0	0	0	0	0	0
	1	1	1	1	0	1	1	1	0	0	1	1	1	0	1	1
	83	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0
	84	1	0	1	0	1	0	0	1	0	1	1	1	0	1	1
	97	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0
3	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0
	1	1	0	1	0	1	0	0	1	0	1	1	1	0	1	1
	83	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0
4	0	0	1	0	0	1	1	0	0	0	0	0	0	0	0	0
	1	1	1	0	0	1	1	1	0	0	1	1	1	0	1	1
	83	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0
5	0	0	1	0	1	0	1	0	0	0	0	0	0	0	0	0
	1	1	1	0	1	0	1	1	0	1	0	0	1	1	0	0
	51	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0
	1	1	0	1	1	0	0	0	1	1	0	0	1	1	0	0
	51	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0

Table 4: All the different binarized metabolic steady states of each experiment. They are the input data used to solve the inference problem.

References

1. Baral, C.: Knowledge Representation, Reasoning and Declarative Problem Solving. Cambridge University Press, New York, NY, USA (2003)
2. Bernot, G., Comet, J.P., Richard, A., Guespin, J.: Application of formal methods to biological regulatory networks: extending thomas' asynchronous logical approach with temporal logic. Journal of Theoretical Biology **229**(3), 339–347 (2004). https://doi.org/10.1016/j.jtbi.2004.04.003
3. Buescher, J.M., Liebermeister, W., Jules, M., Uhr, M., Muntel, J., Botella, E., Hessling, B., Kleijn, R.J., Chat, L.L., Lecointe, F., Mader, U., Nicolas, P., Piersma, S., Rugheimer, F., Becher, D., Bessieres, P., Bidnenko, E., Denham, E.L., Dervyn, E., Devine, K.M., Doherty, G., Drulhe, S., Felicori, L., Fogg, M.J., Goelzer, A., Hansen, A., Harwood, C.R., Hecker, M., Hubner, S., Hultschig, C., Jarmer, H., Klipp, E., Leduc, A., Lewis, P., Molina, F., Noirot, P., Peres, S., Pigeonneau, N., Pohl, S., Rasmussen, S., Rinn, B., Schaffer, M., Schnidder, J., Schwikowski, B., Dijl, J.M.V., Veiga, P., Walsh, S., Wilkinson, A.J., Stelling, J., Aymerich, S., Sauer, U.: Global network reorganization during dynamic adaptations of bacillus subtilis metabolism. Science **335**(6072), 1099–1103 (2012). https://doi.org/10.1126/science.1206871
4. Chaves, M., Oyarzún, D.A., Gouzé, J.L.: Analysis of a genetic-metabolic oscillator with piecewise linear models. Journal of Theoretical Biology **462**, 259–269 (2019). https://doi.org/10.1016/j.jtbi.2018.10.026
5. Chaves, M., Tournier, L., Gouzé, J.L.: Comparing boolean and piecewise affine differential models for genetic networks. Acta Biotheor **58**(2-3), 217–232 (2010). https://doi.org/10.1007/s10441-010-9097-6
6. Chevalier, S., Froidevaux, C., Pauleve, L., Zinovyev, A.: Synthesis of boolean networks from biological dynamical constraints using answer-set programming. In: 2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI). IEEE (2019). https://doi.org/10.1109/ictai.2019.00014

7. Covert, M.W., Knight, E.M., Reed, J.L., Herrgard, M.J., Palsson, B.O.: Integrating high-throughput and computational data elucidates bacterial networks. Nature **429**(6987), 92–96 (2004). https://doi.org/10.1038/nature02456
8. Covert, M.W., Palsson, B.Ø.: Transcriptional regulation in constraints-based metabolic models of Escherichia coli. J Biol Chem **277**(31), 28058–28064 (2002). https://doi.org/10.1046/j.1462-2920.2002.00282.x
9. Covert, M.W., h. Schilling, C., Palsson, B.: Regulation of gene expression in flux balance models of metabolism. Journal of Theoretical Biology **213**(1), 73–88 (2001). https://doi.org/10.1006/jtbi.2001.2405
10. Eiter, T., Gottlob, G.: On the computational cost of disjunctive logic programming: Propositional case. Annals of Mathematics and Artificial Intelligence **15**(3-4), 289–323 (sep 1995). https://doi.org/10.1007/bf01536399
11. Eiter, T., Ianni, G., Krennwallner, T.: Answer Set Programming: A Primer, pp. 40–110. Springer Berlin Heidelberg, Berlin, Heidelberg (2009). https://doi.org/10.1007/978-3-642-03754-2_2
12. Gebser, M., Kaminski, R., Kaufmann, B., Schaub, T.: Answer Set Solving in Practice. Synthesis Lectures on Artificial Intelligence and Machine Learning, Morgan and Claypool Publishers (2012)
13. Gebser, M., Kaminski, R., Kaufmann, B., Schaub, T.: Clingo = ASP + control: Preliminary report. CoRR **abs/1405.3694** (2014)
14. Gebser, M., Kaminski, R., Schaub, T.: Complex optimization in answer set programming. Theory and Practice of Logic Programming **11**(4-5), 821–839 (2011). https://doi.org/10.1017/s1471068411000329
15. Gebser, M., Kaufmann, B., Romero, J., Otero, R., Schaub, T., Wanko, P.: Domain-specific heuristics in answer set programming. Proceedings of the AAAI Conference on Artificial Intelligence **27**(1) (Jun 2013), `https://ojs.aaai.org/index.php/AAAI/article/view/8585`
16. de Jong, H.: Modeling and simulation of genetic regulatory systems: A literature review. Journal of Computational Biology **9**, 67–103 (2002). https://doi.org/10.1089/10665270252833208
17. Liu, L., Bockmayr, A.: Regulatory dynamic enzyme-cost flux balance analysis: A unifying framework for constraint-based modeling. Journal of Theoretical Biology **501**, 110317 (2020). https://doi.org/10.1016/j.jtbi.2020.110317
18. Marmiesse, L., Peyraud, R., Cottret, L.: FlexFlux: combining metabolic flux and regulatory network analyses. BMC Systems Biology **9**(1) (2015). https://doi.org/10.1186/s12918-015-0238-z
19. Orth, J.D., Thiele, I., Palsson, B.Ø.: What is flux balance analysis? Nat Biotechnol **28**(3), 245–248 (2010). https://doi.org/10.1038/nbt.1614
20. Ostrowski, M., Paulevé, L., Schaub, T., Siegel, A., Guziolowski, C.: Boolean network identification from perturbation time series data combining dynamics abstraction and logic programming. Biosystems **149**, 139–153 (2016). https://doi.org/10.1016/j.biosystems.2016.07.009
21. Oyarzún, D.A., Chaves, M., Hoff-Hoffmeyer-Zlotnik, M.: Multistability and oscillations in genetic control of metabolism. Journal of Theoretical Biology **295**, 139–153 (2012). https://doi.org/10.1016/j.jtbi.2011.11.017
22. Razzaq, M., Paulevé, L., Siegel, A., Saez-Rodriguez, J., Bourdon, J., Guziolowski, C.: Computational discovery of dynamic cell line specific boolean networks from multiplex time-course data. PLOS Computational Biology **14**(10), e1006538 (2018). https://doi.org/10.1371/journal.pcbi.1006538

23. Saez-Rodriguez, J., Alexopoulos, L.G., Epperlein, J., Samaga, R., Lauffenburger, D.A., Klamt, S., Sorger, P.K.: Discrete logic modelling as a means to link protein signalling networks with functional analysis of mammalian signal transduction. Mol Syst Biol **5**(1), 331 (2009). https://doi.org/10.1038/msb.2009.87
24. Tournier, L., Goelzer, A., Fromion, V.: Optimal resource allocation enables mathematical exploration of microbial metabolic configurations. J. Math. Biol. **75**(6-7), 1349–1380 (2017). https://doi.org/10.1007/s00285-017-1118-5
25. Tsiantis, N., Balsa-Canto, E., Banga, J.R.: Optimality and identification of dynamic models in systems biology: an inverse optimal control framework. Bioinformatics **34**(14), 2433–2440 (2018). https://doi.org/10.1093/bioinformatics/bty139
26. Videla, S., Saez-Rodriguez, J., Guziolowski, C., Siegel, A.: caspo: a toolbox for automated reasoning on the response of logical signaling networks families. Bioinformatics p. btw738 (2017). https://doi.org/10.1093/bioinformatics/btw738
27. Zañudo, J.G.T., Yang, G., Albert, R.: Structure-based control of complex networks with nonlinear dynamics. Proc Natl Acad Sci USA **114**(28), 7234–7239 (2017). https://doi.org/10.1073/pnas.1617387114

IV MERRIN: a Dedicated Hybrid Solving Framework for the Flux-Based Inference Problem

In this chapter, we introduce a novel workflow, and its implementation *MERRIN*, to solve the flux-based formulation of the inference problem defined in Chapter II. The content of this chapter has been presented at the *European Conference on Computational Biology* (ECCB) of 2022 and the associated paper published in *Bioinformatics* (Thuillier et al., 2022).

To sum up

This work on MERRIN is a first step toward an efficient framework to infer metabolic regulatory rules from time series data. MERRIN's framework relies on monotone properties over optimal metabolic fluxes to efficiently explore the space of all candidate Boolean networks. Our results on a model of core-carbon metabolism are promising, and show that inferring is possible solely from noisy kinetics and transcriptomics time series. However, MERRIN exhibits scalability issues on larger instances. It is, therefore, necessary to further optimize the inferring framework to handle medium-scale regulated metabolic networks.

In this chapter

1 Problem Statement

Relaxed formulation problem's limitations. In Chapter III, we introduce a Boolean relaxation of the inference problem. Addressing the inference problem by solving this Boolean relaxation has shown two main drawbacks: **(i)** results are highly dependent on the input Boolean objective function, and **(ii)** false positive Boolean networks (BN) are inferred. The latter drawback can be overcome by enumerating all solutions and then filtering them with rFBA simulations. However, this post-processed filtering operation is intractable in practice when thousands of solutions should be checked.

Those two drawbacks come from the definition of the Boolean metabolic steady-states (BMSS) used to model the metabolism dynamics. This definition does not allow quantifying metabolic activities and generates spurious BMSSs, *i.e.* BMSSs for which no real-valued counterparts are satisfying the FBA equations. To refine this definition, it is necessary to ensure that only non-spurious BMSSs are used during the inferring process.

Flux-based inference problem. From this statement, we deduce that it is necessary to find a novel inferring framework that handles the FBA equations directly during the solving process. In this chapter, we consider the flux-based formulation of the inference problem (described in Chapter II). Unlike the relaxed formulation, the flux-based formulation relies on the rFBA to model the regulated metabolic network (RMN) dynamics. Regulated metabolic steady-states (RMSS) are used

in place of BMSSs to model the RMN states. The flux-based inference problem formulation is recalled below.

Flux-based formulation of the inference problem

Input:

1: a metabolic network $\mathcal{N} = (\mathcal{M}_{\text{ext}} \cup \mathcal{M}_{\text{int}}, \mathcal{R}, s)$ with an objective reaction 'growth';
2: a set of regulatory proteins $\mathcal{P}$;
3: a set of observed time series $\{\mathcal{T}_o^1, \cdots, \mathcal{T}_o^q\}$, $q \geq 1$;
4: a prior knowledge network $\mathcal{G}$ of dimension $n = |\mathcal{P}| + |\mathcal{M}_{\text{ext}}| + |\mathcal{R}|$;
5: a maximum distance $K_{\max} \in \mathbb{N}$ between observations;
6: a noise rate parameter $\epsilon \in [0, 1[$.

Output: $\underset{f \in \mathbb{F}(\mathcal{G})}{\arg\min} \sum_{k=1}^{q} l_k$

such that:

$$\forall\, \mathcal{T}_k \in \{\mathcal{T}_1, \cdots, \mathcal{T}_q\},\ \exists\, \{(v^j, w^j, x^j)\}_{j=1}^{l_k} \in \text{rFBA}(\mathcal{N}, \mathcal{P}, f),\ \forall\, 1 \leq i < |\mathcal{T}_k|,$$

$$|\mathcal{T}_k| \leq l_k \leq |\mathcal{T}_k| + K_{\max} \tag{IV.1a}$$

$$\wedge\ 0 < g_k(i+1) - g_k(i) \tag{IV.1b}$$

$$\wedge\ (v^{g_k(i)}, w^{g_k(i)}, x^{g_k(i)}) \text{ and } (\hat{v}^i, \hat{w}^i, \hat{x}^i) \text{ are data-compatible} \tag{IV.1c}$$

$$\wedge\ \frac{\hat{v}^i_{\text{growth}}}{1+\epsilon} \leq v^{g_k(i)}_{\text{growth}}\ \wedge \max_{v' \in \text{rMSS}(\mathcal{N}, w^{g_k(i)}, x^{g_k(i)})} v'_{\text{growth}} \leq \frac{\hat{v}^i_{\text{growth}}}{1-\epsilon} \tag{IV.1d}$$

where $g_k : [0, |\mathcal{T}_k|] \to [0, l_k]$ is a bijective function mapping observations of the observed time series $\mathcal{T}_k$ to RMSSs of the trace $\{(v^j, w^j, x^j)\}_{j=1}^{l_k}$.

In practice, we only solve the inference problem for subset-minimal and locally monotone BNs. The subset-minimal criterion is defined according to a partial ordering of BNs on the disjunctive normal form (DNF) of the local functions. However, the problem definition and the solving framework do not rely on these assumptions and can enumerate all solution BNs.

Questions. Therefore, new questions arise: *How to solve hybrid problems merging logical and linear constraints? How to efficiently ensure the satisfiability of the FBA equations during the solving process? How to benchmark an inferring method and test its robustness?*

We aim to answer these questions through the work presented in this chapter. An overview of the main contributions of Thuillier et al. (2022) is provided in the

next sections: the MERRIN's framework in Sec. 2.1, and the time series generation workflow in Sec. 2.2. Section 3 extends the paper's discussion with new results on the framework scalability.

2 Contributions of ECCB's Paper

2.1 MERRIN: a Hybrid Inferring Framework

Overview of the inferring framework described in `Section 2.3` *of the paper.*

Challenge. The flux-based inference problem is a hybrid optimization problem with logic and linear constraints, and linear constraints over the optimal values of linear systems. In other words, it is formulated as a Satisfiability Modulo Theory (SMT) problem (Barrett and Tinelli, 2018). There exist different solvers that can solve SMT problems, *e.g. z3* (De Moura and Bjørner, 2008). However, it has been shown that these solvers are not efficient in solving highly combinatorial problems (Gebser et al., 2014). This result has been validated during our experiments: it took 10 times more time to solve the combinatorial part of the inference problems with *z3* than with the ASP solver *clingo* (Gebser et al., 2017). Moreover, SMT solvers do not always handle both optimization constraints and quantifiers, which is necessary for us.

ASP-based approaches have been widely used in systems biology, and ASP modulo theory solvers have already been applied to solve ASP modulo quantifier-free linear arithmetics problems, *e.g.* for metabolic network completion (Frioux et al., 2019) and elementary fluxes modes enumeration (Mahout et al., 2020). However, these hybrid ASP solvers cannot be used to solve the inference problem since they do not handle constraints over the optimal value of linear systems, *i.e.* the constraints guaranteeing the phenotype compatibility between an RMSS and an observation (Eq. IV.1d).

> ***Challenge***
> Defining an ASP-based framework that extends ASP with the FBA equations and linear constraints over the FBA optimal values. The approach should be scalable and efficient.

Constraint propagation. We propose to rely on the constraint propagation principle (Clarke et al., 2003; Janhunen et al., 2017) to define our framework. The solving process is split among two solvers: a combinatorial solver, that will ensure that the inferred BNs' dynamics match the observations, and a linear solver used to ensure that the generated BMSSs satisfy the FBA equations and the constraints

over optimal growth. For each candidate BN satisfying the combinatorial part, the linear part is checked. If it succeeds then the candidate is accepted. If it fails then the candidate is a counter-example and is rejected. For each counter-example, new constraints are added to the combinatorial part to prevent generating spurious BMSSs.

Constraint generation. The new constraints are generated based on a monotone property over optimal metabolic fluxes (`equations 5 and 6 in the paper`). In short, the property allows for estimating the optimal growth given sets of inhibited reactions, and therefore filtering all sets of inhibited reactions that will surely not match the observations. Given a set of inhibited reactions, if the predicted optimum growth is over (*resp.* under) the observed growth then all subsets (*resp.* supersets) of inhibited reactions will also have optimal growth values over (*resp.* under) the observation.

This property is essential for the framework's scalability. For the complete instance described in the paper, no results were found in about 2 hours without it while all of them are enumerated under 30 seconds with it.

> ***Solution*** **– in short**
> We propose a hybrid solving framework relying on the constraint propagation principle. The counter-examples are generalized using a monotone property on the optimal metabolic fluxes that filter spurious BMSSs. The monotone property is essential for the method's scalability.

MERRIN. This hybrid framework has been implemented into the tool *MERRIN*[1]. *MERRIN* extends the ASP program used to solve the relaxed inference problem, introduced in Chapter III, with the FBA equations. The combinatorial part of the *MERRIN* framework is identical to the ASP program used for solving the relaxed inference problem, except that it does not have saturation constraints. The FBA equations and the phenotype compatibility are checked dynamically for each BMSS inferred during the solving process. The ASP encoding of the combinatorial part of the *MERRIN* framework is described in Appendix B.3.2.

2.2 Time Series Generation Workflow

Overview of the time series generation workflow described in `Section 3.2` *of the paper.*

[1] Available on GitHub: https://github.com/bioasp/merrin

Challenge. The inferring methods defined, the question of *how to validate it* arises. Ideally, the inferring framework should be validated on real experimental observations. However, benchmarking with experimental data has some limits. Even for model bacteria, such as *Escherichia coli*, regulatory rules are not well known. It is therefore hard to build the input prior knowledge network. Even if such data were easily available, the most up-to-date regulated metabolic models of *Escherichia coli* account for more than one thousand regulatory rules (Covert et al., 2004). It will therefore be hard to explain *MERRIN*'s behaviors (why a regulatory rule has been inferred or not) with models of such size. Another limit is that dynamic experimental observations are often not publicly available, nor provided with papers.

We choose to validate *MERRIN* on the model of core-carbon metabolism introduced in Covert et al. (2001). It is a synthetic regulated metabolic network that reproduces interesting growth behaviors, such as diauxic shifts or anaerobic growth. Relying on a synthetic model has some advantages: the ground truth regulatory network is known, and we can generate our own time series observations while controlling data types and noise rates.

> ***Challenge***
> Generating realistic time series observations that mimic *in vitro* kinetics, fluxomics and/or transcriptomics observations.

Dynamic time series generation. Since we rely on a synthetic model to validate *MERRIN*, we need to generate time series observations. These observations should be as similar as possible to *in vitro* kinetics, fluxomics, and/or transcriptomics observations. Generating realistic observations is necessary to ensure the ability of *MERRIN* to learn regulatory rules from real experimental observations.

Alongside *MERRIN*'s workflow, we introduce in the paper a time series generation protocol. This protocol has been developed through constant discussions with biologists to ensure that the resulting time series are as realistic as possible. It allows converting rFBA simulations into noisy fluxomics, kinetics, and/or transcriptomics observations. The idea is to, first, simplify each rFBA simulation into a few measured time points, approximately 2 observations by metabolic growth phases. Then, the different observations are extracted from the RMSSs of the simplified rFBA simulations: **(i)** for fluxomics observations, the metabolic fluxes of the reactions are kept; **(ii)** for kinetics observations that's the concentration of environmental metabolites; and **(iii)** for transcriptomics observations that's the genes and regulatory proteins states, as well as the availability states and activity states of external metabolites and reactions, respectively. The observations of elements not covered by one of the selected data types are set to an undefined

Data types	Notation	Definition from an RMSS (v, w, x)
Fluxomics	$\hat{v}$	v
Kinetics	$\hat{w}$	w
Transcriptomics	$\hat{x}$	$x_{\mathcal{P}} \cup \beta(w) \cup \beta(v)$

Table 5 – Structure of the observations for each data type from a regulated metabolic steady-state (v, w, x), with $\mathcal{N} = (\mathcal{M}_{\text{ext}} \cup \mathcal{M}_{\text{int}}, \mathcal{R}, s)$ a metabolic network, and $\mathcal{P}$ a set of genes and regulatory proteins. $\beta : \mathbb{R}^n \to \mathbb{B}^n$ is a binarization function such that $\forall\, s \in \mathbb{R}^n$, $\forall\, 1 \leq i \leq n$, $\beta(s)_i = 1$ if and only if $s_i \neq 0$, else $\beta(s)_i = 0$.

value ('$\perp$'), *e.g.* if fluxomics is not selected, then all metabolic flux of reactions are set to '$\perp$'. Table 5 described for each data type how an observation $(\hat{v}, \hat{w}, \hat{x})$ is built from an RMSS (v, w, x).

The time series generation protocol also introduces noise in the data by either altering the quantitative observations by a random amount, removing observations, or removing timesteps of the time series. The benchmark generation protocol is described in further detail in `Sec. 3.2.2` and `Fig. 1b of the paper`.

Solution – **in short**
We use a synthetic regulated metabolic model to validate *MERRIN*, which offers control over the benchmarking process (including noise rate and observation types) and provides a known ground truth BN. We propose a data generation workflow to convert rFBA simulations into realistic kinetics, fluxomics, and transcriptomics time series observations with different noise rates.

Benchmarking The validation of *MERRIN* has been made on a benchmark of 240 instances that combine different data types, namely kinetics, fluxomics, and/or transcriptomics, with noise levels ranging from 0 to 50 percent. Following a parsimonious approach, the benchmarking process focuses only on the subset-minimal BNs inferred by *MERRIN*. Results over the benchmarks are described `in Sec. 3.3` and `Sec. 3.4 of the paper`.

Solution – **in short**
The quality of inferred BNs is measured according to two criteria: **(i)** their similarity with the ground truth BN using accuracy and recall scores; **(ii)** their ability to reproduce the rFBA simulations used to generate the benchmark.

On complete instances, *MERRIN* infers a smaller BN than the ground truth. All the inferred regulatory rules are in the ground truth BN (*accuracy = 1*), but not

all regulatory rules are retrieved (*recall = 0.64*). The associated RMN reproduces exactly the rFBA simulations, which shows that the missing rules are not necessary to explain the input observations regarding the rFBA formalism. *MERRIN* infers this BN when there are at least kinetics and transcriptomics observations with a noise of up to 20 percent.

Moreover, when using only transcriptomics observations, *MERRIN* retrieves all but one regulatory rule of this subset minimal solution. The inferred BN reproduces exactly 4 of the 5 rFBA simulations and differs from the last simulation on only one timestep (*residual sum of square < 1*).

3 Complementary Benchmarking and Discussion

This section extends the discussion of the paper. We introduce new results on MERRIN's scalability that were generated after the paper's publication.

3.1 Summary of ECCB's Paper

In this chapter, we have introduced a flux-based formulation of the inference problem and a hybrid solving framework to solve it. This framework has been implemented into the tool *MERRIN*. It relies on the ASP program used to solve the relaxed inference problem of Chapter III. By extending the ASP solver *clingo* with the FBA, we ensure that no spurious BMSSs are generated and that the optimal growths, allowed by regulatory states, are compatible with the observed growth phenotypes.

To validate *MERRIN*, we define a time series generation protocol that allows generating realistic synthetic time series observations from rFBA simulations. Using this time series generation protocol, we generate a benchmark of 240 instances of noisy time series of kinetics, fluxomics, and/or transcriptomics data of a core-carbon metabolism model. Our results suggest that it is possible to infer regulatory rules, including feedback and control rules, from solely kinetics and transcriptomics observations with up to 20% of noise.

3.2 MERRIN's Performance on Small-Scale Instances.

Table 6 summarizes the number of inferred BNs and the computation times[2] of *MERRIN* on three RMNs: the *toy* model introduced in Chapter III, the model of *core*-carbon metabolism (Covert et al., 2001) used in the paper, and a *medium*-scale model of *Escherichia coli* core metabolism (Covert and Palsson, 2002). All results

[2]Fedora 34 with an 8-cores processor i7-1165G7@2.80 GHz and 16GB of RAM

Instance	Search space size	All networks #	All networks Time (s)	Subset-minimal networks #	Subset-minimal networks Time (s)
Toy	$O(10^6)$	4	1.5	1	1
Core	$O(10^{15})$	48	30	1	7
Medium-scale	$\Omega(10^{380})$	-*	$> 86\,400$*	-*	$> 86\,400$*

* *No Boolean networks inferred in 24 hours.*

Table 6 – Performances of *MERRIN* on three regulated metabolic networks (described in Chapter II). Times are given as an approximated order of magnitude in seconds. All results are given for complete data (undegraded kinetics, fluxomics, and transcriptomics time series observations).

discussed in this section are for complete time series observations, *i.e.* noise-free time series with kinetics, fluxomics, and transcriptomics observations.

Exact resolution of the inference problem. The flux-based inference problem allows for inferring BNs exactly compatible with the rFBA dynamics. On the *toy* model, *MERRIN* infers 4 BNs, of which one is subset-minimal. For the *core* model, *MERRIN* infers 48 BNs, of which one is subset-minimal. It takes less than 30 seconds to infer the 48 BNs, and about 7 seconds to only enumerate the subset-minimal ones. The RMN associated with each inferred subset-minimal BN reproduces exactly the rFBA simulations used to generate the input time series.

The flux-based definition, and the hybrid solving framework used to solve it, allow, therefore, overcoming the drawbacks of the relaxed inference problem. No false-positive solutions are inferred, and there is no need for a Boolean objective function. For the latter, it is replaced by the 'growth' reaction commonly used in FBA-based frameworks.

Performance. *MERRIN* performances lie in the constraint generation method used to filter spurious candidate BNs. The constraint generation allows enumerating all solutions of the flux-based inference problem in under 30 seconds, while no solutions were inferred in 2 hours without it.

Limits of the rFBA formalism. On the *core* model, it must be noted that not all the ground truth regulatory rules are recovered in the subset-minimal BN. The subset-minimal BN misses 4 ground truth regulatory rules, while still being able to reproduce exactly the input rFBA simulations. While we explain why they are not

recovered in the paper, our conclusions highlight the limit of the rFBA formalism to model the RMN dynamics.

Recall that the FBA relies on two heuristics: **(i)** that biomass is maximized and **(ii)** that the metabolism is at a steady state. Therefore, the FBA implicitly handles some regulatory rules, there is no need for Boolean regulatory rules to model them. Since *MERRIN* relies on the rFBA for the inferring of metabolic regulatory rules, it will not be able to learn regulatory rules that are directly handled by the FBA.

3.3 Limitation: Scalability of MERRIN on Larger Instances

While not being part of this publication, the scalability of *MERRIN* to larger RMNs has been tested. In particular, this was a question raised by the reviewers when the paper was submitted.

Medium-scale instance. We apply *MERRIN* to the *medium*-scale model of *Escherichia coli* core metabolism introduced in Covert and Palsson (2002). This medium-scale model has about 5 times more reactions and 15 times more regulatory rules than the *core* model. A comprehensive description of the medium-scale model is given in Appendix A.2. The instance of the inference problem was built following the protocol defined in the paper from the three experimental conditions provided in the aforementioned paper. The instance is composed of three noise-free times series of kinetics, fluxomics, and transcriptomics observations. The prior knowledge network models a search space compatible with about 10^{380} BNs.

Results. As described in Table 6, *MERRIN* was not able to infer any BNs under 24 hours. The threshold of 24h is an arbitrary choice and was defined as a reasonable time limit to infer at least one BN. Regarding this result, there is a need to further improve the solving process to allow inferring larger BNs controlling metabolic networks.

Bioinformatics, 38, 2022, ii127–ii133
https://doi.org/10.1093/bioinformatics/btac479
Proceedings of ECCB2022

OXFORD

Systems Biology

MERRIN: MEtabolic Regulation Rule INference from time series data

Kerian Thuillier [1]*, Caroline Baroukh [2], Alexander Bockmayr [3], Ludovic Cottret [2], Loïc Paulevé [4], Anne Siegel [1]*

[1]Univ Rennes, Inria, CNRS, IRISA, F-35000 Rennes, France
[2]LIPME, INRAE, CNRS, Université de Toulouse, Castanet-Tolosan, France
[3]Freie Universität Berlin, Institute of Mathematics, D-14195 Berlin, Germany
[4]Univ. Bordeaux, Bordeaux INP, CNRS, LaBRI, UMR5800, F-33400 Talence, France

*To whom correspondence should be addressed.

Abstract

Motivation: Many techniques have been developed to infer Boolean regulations from a prior knowledge network and experimental data. Existing methods are able to reverse-engineer Boolean regulations for transcriptional and signaling networks, but they fail to infer regulations that control metabolic networks.
Results: We present a novel approach to infer Boolean rules for metabolic regulation from time series data and a prior knowledge network. Our method is based on a combination of answer set programming and linear programming. By solving both combinatorial and linear arithmetic constraints we generate candidate Boolean regulations that can reproduce the given data when coupled to the metabolic network. We evaluate our approach on a core regulated metabolic network and show how the quality of the predictions depends on the available kinetic, fluxomics or transcriptomics time series data.
Availability: Software available at https://github.com/bioasp/merrin
Contact: anne.siegel@irisa.fr
Supplementary information: See supplementary PDF and https://doi.org/10.5281/zenodo.6670165

1 Introduction

The regulation of metabolic gene expression is essential for an organism to respond appropriately to changes in its environment. For three decades now, methods have been developed to model, simulate and infer gene regulatory networks (de Jong, 2002; Bernot *et al.*, 2004; Chaves *et al.*, 2010). Even with the advances of next generation -omics, such networks remain largely incomplete and unable to accurately predict complex responses of organisms submitted to changes in diverse environments.

The methods developed so far to infer Boolean dynamics of regulatory and signaling networks only rely on information on the regulatory layer of the cell, mainly transcriptomics, proteomics and phosphoproteomics (Saez-Rodriguez *et al.*, 2009; Videla *et al.*, 2017; Razzaq *et al.*, 2018; Tsiantis *et al.*, 2018; Chevalier *et al.*, 2019). However, studying the metabolic layer could help to better infer the regulatory rules. Catabolic repression is a good illustration of how metabolism can highlight regulations inside the cell. This happens when the cell first consumes one substrate (e.g. hexose) until it is exhausted before starting to consume other substrates present in the environment (Monod, 1942). Looking only at the metabolites in the environment, we can infer that a regulation takes place inside the cell, probably on transporters.

Up to now, very few approaches exploited the metabolic layer of the organism to obtain regulatory information. In (Tournier *et al.*, 2017), Resource Balance Analysis (RBA) (Goelzer *et al.*, 2015) is used to infer logical rules governing the activation of metabolic fluxes in response to diverse extracellular media. However, the authors assume that no feedback from metabolism to regulation occurs, which does not correspond to the biological functioning of the cell in most cases.

The fact that metabolic and regulatory layers are of different nature, and thus formalized differently, makes the inference of regulations challenging. The metabolic layer is usually modeled by a metabolic network consisting of a weighted hypergraph with metabolites as nodes, reactions as hyperarcs, and stoichiometry as weights. The (dynamic) response of the metabolism to the environment is usually modeled by Flux Balance Analysis (FBA) (Orth *et al.*, 2010) resp. dynamic FBA (dFBA) (Mahadevan *et al.*, 2002). This approach assumes that the metabolism of the cell is at quasi steady-state and that the cellular behavior is optimal with respect to some objective (usually growth). FBA and dFBA require solving linear programming problems; the output is the

prediction of metabolic fluxes and the concentrations of environmental metabolites and biomass, which are all continuous quantitative data. On the contrary, the dynamics of the regulatory layer is often modeled by Boolean networks (BNs). Combining both layers to infer regulations of the cell and taking into account feedbacks between them thus requires to use a hybrid discrete-continuous modeling and inference framework, such as Satisfiability Modulo Theories (SMT), which was used in Frioux *et al.* (2019) to solve a metabolic network completion problem.

In this study, we present a hybrid discrete-continuous approach to infer metabolic regulations, which combines linear programming for metabolism with answer set programming for regulations. The input consists of a metabolic network, a prior knowledge regulatory network with potential regulations, and time series data. These can be metabolomics data (kinetics of environmental metabolites/biomass and/or fluxomics) and/or expression data from proteomics or transcriptomics. The output is a set of Boolean regulatory networks that best explain the available data. We tested our method on data generated from a dynamic regulatory FBA (d-rFBA) model of a core regulated metabolic network (Covert *et al.*, 2001; Marmiesse *et al.*, 2015), by simulating both the regulatory and the metabolic layer in five environments. In order to assess its robustness, the method was also evaluated with noisy and partial data, *e.g.* transcriptomics and kinetics of environmental metabolites only.

2 Methods and implementation

2.1 d-rFBA: coupling metabolic and regulatory networks

2.1.1 Regulated metabolic networks (RMN), influence graph

A *regulated metabolic network* (RMN) consists of (i) a metabolic layer characterized by linear constraints on metabolic fluxes and (ii) a regulatory layer specified by a Boolean network (BN) which models the interplay between metabolic fluxes, input metabolites, and regulatory proteins.

Formally, a RMN is a quadruple $(\mathcal{N}, \mathrm{Inp}, \mathcal{P}, f)$ composed of (i) a metabolic network $\mathcal{N} = (\mathrm{Int}, \mathrm{Ext}, \mathcal{R}, S)$ with a set of internal metabolites Int, a set of external metabolites Ext, a set of irreversible reactions $\mathcal{R}$ and a stoichiometric matrix $S \in \mathbb{R}^{(|\mathrm{Int}|+|\mathrm{Ext}|)\times|\mathcal{R}|}$. Each reaction $r \in \mathcal{R}$ is associated with flux bounds $l_r, u_r \in \mathbb{R}, 0 \leq l_r \leq u_r$; (ii) a set of input metabolites $\mathrm{Inp} \subseteq \mathrm{Ext}$; (iii) a set of regulatory proteins $\mathcal{P}$; (iv) a BN $f : \mathbb{B}^n \to \mathbb{B}^n, \mathbb{B} = \{0, 1\}$, of dimension $n = |\mathrm{Inp}| + |\mathcal{R}| + |\mathcal{P}|$. We call $f_i : \mathbb{B}^n \to \mathbb{B}$ the *local function* of component i.

The *influence graph* $G(f)$ summarizes the regulatory dependencies. It is a signed directed graph with node set $\mathrm{Inp} \cup \mathcal{R} \cup \mathcal{P}$ and a positive (resp. negative) edge from j to i if there exists $x \in \mathbb{B}^n$ such that an increase of x_j leads to an increase (resp. decrease) of $f_i(x)$. We assume that f is *locally monotone*, *i.e.*, there exists at most one edge from j to i, but our method does not rely on this assumption. In RMNs, the regulation of reactions has to be mediated by regulatory proteins $\mathcal{P}$. Therefore, there is no edge from j to i in $G(f)$ where both $i, j \in \mathrm{Inp} \cup \mathcal{R}$. Edges between regulatory proteins $i, j \in \mathcal{P}$, however, are possible.

2.1.2 Regulatory-metabolic steady states (RMSSs)

Dynamic regulatory Flux Balance Analysis (d-rFBA) (Covert *et al.*, 2001) extends FBA to derive a discrete time series of steady states optimal for a linear objective. In d-rFBA, a *regulatory-metabolic steady state* (RMSS) of a RMN $(\mathcal{N}, \mathrm{Inp}, \mathcal{P}, f)$ is a triple (v, c, x) associating reaction fluxes v at steady state, concentrations c of external metabolites, and the state x of the Boolean network, which comprises the Boolean regulatory state of reactions and regulatory proteins, and the binarization of the concentration of input metabolites. The reaction fluxes v are constrained by both the regulatory variables x, which can force reaction fluxes to be zero, and by the concentration of external metabolites c, which set upper bounds on uptake fluxes. Formally, a RMSS is a triple $(v, c, x) \in \mathbb{R}^{|\mathcal{R}|} \times \mathbb{R}^{|\mathrm{Ext}|} \times \mathbb{B}^{|\mathrm{Inp}|+|\mathcal{R}|+|\mathcal{P}|}$ such that

$(1.a)\ S_{\mathrm{Int},\mathcal{R}} \cdot v = 0, \quad (1.b)\ \forall r \in \mathcal{R}, l_r \cdot x_r \leq v_r \leq u_r \cdot x_r$
$(1.c)\ \forall m \in \mathrm{Inp}, r \in \mathcal{R}, S_{mr} < 0 \Rightarrow v_r \leq \mathrm{uptake\_bound}(c_m),$

where $S_{\mathrm{Int},\mathcal{R}}$ is the submatrix of S whose rows correspond to internal metabolites and $\mathrm{uptake\_bound}(c_m)$ is the maximum flux through uptake reaction r for input metabolite concentration c_m (Varma and Palsson, 1994).

2.1.3 Dynamics of RMNs and admissible time series

The d-rFBA models are executed at two time scales: the metabolic network, considered as a fast system, depending on the activity of input metabolites and regulatory proteins, rapidly converges to a steady state; the regulatory network, considered as a slow system, gets updated once the metabolic network is in steady state. The overall dynamics is guided by the objective of maximizing the flux through reaction $Growth$, assumed to reflect the growth of the cell (Feist and Palsson, 2010).

Let $\beta : \mathbb{R}^n_{\geq 0} \to \mathbb{B}^n$ be a binarization function such that $\forall s \in \mathbb{R}^n_{\geq 0}, \forall i \in \{1, \ldots, n\}, \beta(s)_i = 1$ if and only if $s_i > 0$, else $\beta(s)_i = 0$. Given a RMSS (v^k, c^k, x^k) at time t^k, a successor RMSS $(v^{k+1}, c^{k+1}, x^{k+1})$ at time t^{k+1} is computed as follows:

1. The external metabolite concentrations c^{k+1} are computed from the previous concentrations c^k by considering constant uptake/secretion fluxes v^k for the whole time period $[t^k, t^{k+1}]$.
2. The Boolean state x^{k+1} is computed by applying the regulatory function f to the binarized input metabolites concentrations $x'_{\mathrm{Inp}} = \beta(c^{k+1}_{\mathrm{Inp}})$ at time t^{k+1}, together with the binarized reaction fluxes $x'_{\mathcal{R}} = \beta(v^k)$ and the Boolean values $x'_{\mathcal{P}} = x^k_{\mathcal{P}}$ of the regulatory proteins at time t^k, *i.e.*, $x^{k+1} = f(x')$.
3. $(v^{k+1}, c^{k+1}, x^{k+1})$ is a RMSS maximizing the flux through the $Growth$ reaction, *i.e.*, there is no RMSS (v', c^{k+1}, x^{k+1}) such that $v'_{Growth} > v^{k+1}_{Growth}$.

Such simulations can be computed with the FlexFlux implementation of d-rFBA (Marmiesse *et al.*, 2015), which considers a fixed time step τ between successive RMSS, see Thuillier *et al.* (2021) for details.

Let $\mathbb{S}$ be the set of all RMSSs of the RMN $(\mathcal{N}, \mathrm{Inp}, \mathcal{P}, f)$. For input metabolite concentrations $c_0 \in \mathbb{R}^{|\mathrm{Ext}|}$ and the regulatory state $x_0 \in \mathbb{B}^{|\mathrm{Inp}|+|\mathcal{P}|+|\mathcal{R}|}$, we denote by $\max_{Growth} \mathrm{rMSS}(c_0, x_0) = \max\{v_{Growth} \mid (v, c_0, x_0) \in \mathbb{S}\}$ the maximum growth flux given c_0 and x_0. Given reaction fluxes $v, v' \in \mathbb{R}^{|\mathcal{R}|}$, external metabolite concentrations $c, c' \in \mathbb{R}^{|\mathrm{Ext}|}$, and regulatory states $x, x' \in \mathbb{B}^{|\mathrm{Inp}|+|\mathcal{R}|+|\mathcal{P}|}$, d-rFBA enables a transition from (v, c, x) to (v', c', x') if and only if the following constraints are satisfied:

$(2.a)\ c' = \mathrm{update}(c, v), \quad (2.b)\ x' = f(\beta(c'_{\mathrm{Inp}}), \beta(v), x_{\mathcal{P}}),$
$(2.c)\ (v', c', x') \in \mathbb{S}, \quad (2.d)\ v'_{Growth} = \max_{Growth} \mathrm{rMSS}(c', x'),$

where $\mathrm{update}(c, v)$ updates the external metabolite concentrations c according to reaction fluxes, stoichiometry, and cell volume changes. Eq.(2.c) encompasses Eqs.(1.a-c). As shown in Thuillier *et al.* (2021), one can derive a necessary Boolean condition for these constraints (see Suppl. Sect. 2), which we denote by Eq.($2.c_{\mathrm{relaxed}}$).

2.2 The inference problem for regulatory rules

Next we address the compatibility between the d-rFBA dynamics of a RMN and given time series data for reaction fluxes, regulatory protein states and input metabolite concentrations.

Observed time series. An *observation* is a triple $o = (v_{Growth}, c, x_{\mathcal{P}})$, where (i) $v_{Growth} \in \mathbb{R}$ denotes a *Growth* flux, (ii) $c \in \mathbb{R}^{|\mathrm{Inp}|}$ the input metabolite concentrations, (iii) $x_{\mathcal{P}} \in (\mathbb{B} \cup \{\perp\})^{|\mathcal{P}|}$ represents regulatory protein states, which can be either Boolean values or undefined

("$\perp$"). An *observed time series* is a sequence of observations $T_O = (o_0, \cdots, o_m), m \geq 0$.

Compatibility between an observed time series and a RMN. A RMN and an observed time series $T_O = (o_0, \cdots, o_m)$, with $o_i = (v_{Growth_i}, c_i, x_{\mathcal{P}_i}), 0 \leq i \leq m$, are said to be *compatible with maximum distance* $K \in \mathbb{N}$ *and noise rate* $0 \leq \epsilon < 1$ if there exists a d-rFBA simulation $T_S = (\hat{s}_0, \ldots, \hat{s}_l), l \geq m$, of the RMN, with RMSS $\hat{s}_j = (\hat{v}_j, \hat{c}_j, \hat{x}_j), 0 \leq j \leq l$, and a function $g : \{0, \ldots, m\} \to \{0, \ldots, l\}$ associating each observation with a RMSS, such that the following conditions are satisfied for $0 \leq i \leq m$:

$(3.a)\ 0 < g(i+1) - g(i) \leq K, \qquad (3.b)\ \hat{x}_{g(i)_{\mathrm{Inp}}} = \beta(c_i),$

$(3.c)\ \forall p \in \mathcal{P}, x_{i_p} \neq \perp \implies \hat{x}_{g(i)_p} = x_{i_p},$

$(3.d)\ \frac{v_{Growth_i}}{1+\epsilon} \leq \max_{Growth} \mathrm{rMSS}(c_i, \hat{x}_{g(i)}) \leq \frac{v_{Growth_i}}{1-\epsilon}.$

Eq.(3.a) states that consecutive observations are separated by at most K d-rFBA simulation steps. Eq.(3.b) ensures the complete match between the discretized values of the d-rFBA simulation and the observed inputs. Eq.(3.c) constrains the Boolean states of proteins in the d-rFBA simulation to be equal to the observed ones, when available. Eq.(3.d) states that the simulated growth is close (up to the allowed noise) to the observed growth.

Inference problem. Eqs.(2) in Sect. 2.1.3 characterize the admissible sequences of RMSSs w.r.t. a given RMN and Eqs.(3) the compatibility between a RMN and an observed time series. The problem of inferring regulatory rules compatible with a set of observed time series is:

Problem statement tackled by MERRIN: Inferring regulatory rules from observed time series

Input:

1: a set of observed time series $\{T^1, \cdots, T^q\}, q \geq 1$;
2: a metabolic network $\mathcal{N} = (\mathrm{Int}, \mathrm{Ext}, \mathcal{R}, S)$;
3: a set of regulatory proteins $\mathcal{P}$;
4: a prior knowledge network (PKN) $\mathcal{G}$ whose nodes belong to $\mathrm{Inp} \cup \mathcal{P} \cup \mathcal{R}$ and such that there is no $i \xrightarrow{s} j \in \mathcal{G}$ with $i, j \in \mathrm{Inp} \cup \mathcal{R}$;
5: a noise parameter $\epsilon \in [0, 1[$;
6: a maximum distance $K \in \mathbb{N}$ between observations.

Output: All BNs $f \in \mathbb{B}^{|\mathrm{Inp}|+|\mathcal{R}|+|\mathcal{P}|}$ such that:

1: f is locally monotone;
2: $G(f) \subseteq \mathcal{G}$;
3: for each T^i the associated RMN $(\mathcal{N}, \mathrm{Inp}, \mathcal{P}, f)$ has a d-rFBA simulation T_S compatible with T^i (satisfying Eqs.(3));
4: there is no BN $f' \in \mathbb{F}$ smaller than f considering the local functions in disjunctive normal form (subset minimality ordering).

In practice, we focus on the *smallest* (subset-minimal) compatible BNs by considering a partial ordering between BNs based on the disjunctive normal form (DNF) of the local functions (Chevalier *et al.*, 2019). However, our approach can be used to enumerate all compatible BNs, not only the subset-minimal ones.

2.3 Resolution using hybrid Answer Set Programming

The inference problem relies on hybrid optimization as it requires exploring the combinatorial domain of putative regulatory BNs constrained by the PKN, and checking both combinatorial constraints linking consecutive states of regulatory proteins according to a given observed time series (Eq.(2.b) and Eqs.(3.b-c)) and linear arithmetic constraints related to the characterization of RMSSs and v_{Growth} optimization (Eqs.(1), Eqs.(2.c-d), Eq.(3.d)). To solve this problem, we used SMT (Satisfiability Modulo Theory) solving (Barrett and Tinelli, 2018; Janhunen *et al.*, 2017), by implementing a resolution framework relying on constraint propagation: whenever a solution satisfying the combinatorial part is found, the linear part is checked. If the linear check succeeds then the solution is accepted. If it fails then the solution is rejected and new constraints are added to the combinatorial part to avoid alternative solutions which would for sure fail the linear check as well.

Algorithm 1 Hybrid Resolution: $T = \{T^1, \cdots, T^q\}, \mathcal{N}, P, \mathcal{G}, \epsilon, K$

$\mathrm{Inp} \leftarrow \{m \mid m \in \mathrm{Ext}, \exists r \in \mathcal{R}, S_{mr} > 0\}$
$n \leftarrow |\mathrm{Inp}| + |\mathcal{R}| + |\mathcal{P}|$
$\mathbb{F} \leftarrow \{f \mid f \in \mathbb{B}^n \to \mathbb{B}^n, G(f) \subseteq \mathcal{G} \wedge f \text{ is locally monotone}\}$
[ASP solving]
select $\hat{f} \in \mathbb{F}$ verifying (2.a), (2.b) and (2.c$_{\text{relaxed}}$)
$\mathcal{RMN} \leftarrow (\mathcal{N}, \mathrm{Inp}, \mathcal{P}, \hat{f})$
for all $T^i \in T$ **do**
  select a family of RMSS $\{\hat{s}^i_0, \cdots, \hat{s}^i_{l_i}\}$ of the $\mathcal{RMN}$ satisfying constraints (3.a), (3.b) and (3.c)
end for
[Linear solving]
check with linear programming whether (2.c) and (3.d) hold
if (2.c) and (3.d) hold **then**
  $\hat{f}$ is a solution
else
  for all o^i_j and its associated RMSS $\hat{s}^i_k$ **do**
    $o^i_j = (v^i_{Growth_j}, c^i_j, x^i_j)$ and $\hat{s}^i_k = (\hat{v}^i_k, \hat{c}^i_k, \hat{x}^i_k)$
    if $\hat{v}^i_{Growth_k} > (v^i_{Growth_j})/(1-\epsilon)$ **then**
      add Eq.(4) with $x = \hat{x}^i_k$
      exclude any RMSS associated with o^i_j that do not verify Eq.(4).
    else if $\hat{v}^i_{Growth_k} < (v^i_{Growth_j})/(1+\epsilon)$ **then**
      add Eq.(5) with $x = \hat{x}^i_k$
      exclude any RMSS associated with o^i_j that do not verify Eq.(5)
    end if
  end for
  return to step 4
end if

The inference from purely combinatorial constraints was formulated using Answer Set Programming (ASP) (Baral, 2003; Gebser *et al.*, 2012), a logic programming framework for expressing symbolic satisfiability problems. Modern solvers like Clingo (Gebser *et al.*, 2017) support various reasoning modes, including subset-minimal enumeration. The linear arithmetic constraints were formulated in linear programming.

The constraint propagation exploits a monotonicity property of the objective v_{Growth} of RMSSs: for fixed input metabolite concentrations, inhibiting (*resp.* releasing an inhibition of) a reaction cannot increase (*resp.* decrease) the maximum value of v_{Growth}. Thus, given input metabolite concentrations $c_0 \in \mathbb{R}^{|\mathrm{Inp}|}$ and an optimal RMSS (v, c_0, x), we can characterize optimal RMSS (v', c_0, x') for which $v'_{Growth} \leq v_{Growth}$ (Eq.(4)) resp. $v'_{Growth} \geq v_{Growth}$ (Eq.(5)) by requiring

$(4)\ \forall r \in \mathcal{R}, x'_r \leq x_r \qquad \text{resp.} \qquad (5)\ \forall r \in \mathcal{R}, x'_r \geq x_r.$

This allows performing constraint propagation during the combinatorial resolution and further reducing the number of linear programming checks.

Algorithm and implementation. The hybrid resolution of the inference problem is detailed in Algorithm 1. For the sake of simplicity, we explain the global solving scheme on the full time series T, although the software implementation extends this algorithm to incomplete time series. In practice, Algorithm 1 is implemented by extending the Clingo solver, using its Python API, with a linear constraint propagator, implemented

with the python PuLP library, and the solver COIN (Forrest *et al.*, 2022). Each problem instance was executed on Fedora 34 with an 8 core processor i7-1165G7@2.80GHz and 16GB of RAM.

3 Results

3.1 MERRIN workflow

The MEtabolic Regulation Rule Inference (MERRIN) software implements the workflow in Fig. 1(a) to infer regulatory rules of a RMN from possibly incomplete and noisy observed time series (Sect. 2.2 and 3.2) using Algorithm 1.

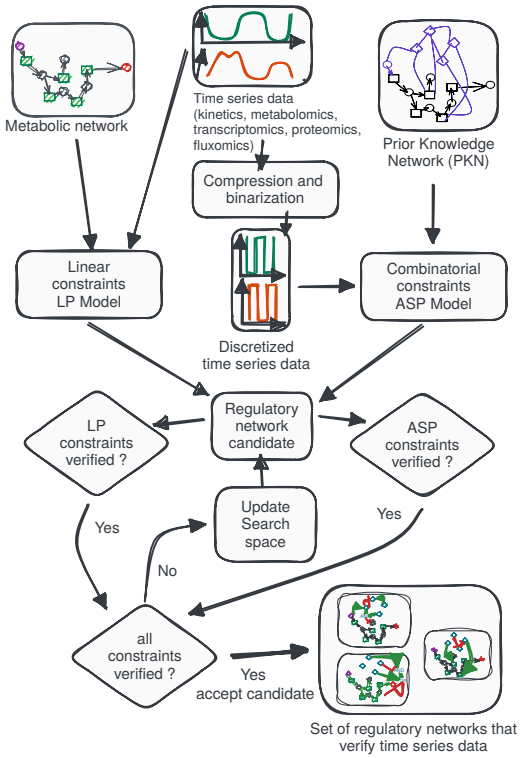
MERRIN takes as *input* (i) a metabolic network $\mathcal{N} = (\mathrm{Int}, \mathrm{Ext}, \mathcal{R}, S)$ in SBML format, (ii) a set of regulatory proteins $\mathcal{P}$ (ii) a set of observed time series $T = \{T^1, \ldots, T^q\}$ with their type (complete, kinetic-fluxomic, kinetic-transcriptomic, transcriptomic) in CSV format, and (iii) a prior knowledge network (PKN) $\mathcal{G}$ in text format. To allow for incomplete and noisy time series, two parameters can be set: (i) $K \in \mathbb{N}$ the maximum number of intermediate unobserved RMSSs for each time series; (ii) $\epsilon \in [0, 1[$ the estimated noise rate. For the rest of the paper, we will consider $\epsilon = 0.3$ and $K = 10$.

The *search space* $\mathbb{F}$ consists of all Boolean networks (BNs) f of dimension $n = |\mathrm{Inp}| + |\mathcal{R}| + |\mathcal{P}|$ whose influence graph $G(f)$ is a subgraph of the PKN $\mathcal{G}$. The size of $\mathbb{F}$ is doubly exponential in n. MERRIN returns as *output* all subset-minimal locally monotone regulatory BNs $f \in \mathbb{F}$ such that the associated RMN $(\mathcal{N}, \mathrm{Inp}, \mathcal{P}, f)$ is compatible with the observed time series $T = \{T^1, \ldots, T^q\}$.

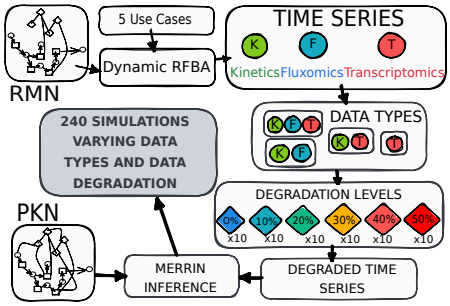
3.2 Application to a core regulated metabolic model

Problem instance. To validate our approach, we applied MERRIN to synthetic data generated for a core regulated metabolic network originally proposed in (Covert *et al.*, 2001), which we refer to as the *gold standard*. (i) The *metabolic layer* of the gold standard (see Fig. 2(a)), also serving as input for MERRIN, contains 20 reactions and 8 external metabolites, among them the 5 inputs Carbon1, Carbon2, Oxygen, Fext, Hext. (ii) The *regulatory layer* of the gold standard involves the four regulatory proteins RPcl, RPO2, RPb, RPh. (iii) In order to explore alternative regulatory rules that could explain the observed time series data, we consider the *PKN* in Fig. 2(b), which includes for each edge in the influence graph of the gold standard all possible combinations of signs and directions. Moreover, two edges from Carbon2 to RPcl, and four edges between RPcl and Tc1 were added as possible alternative regulations to be explored. It follows that the search space to be explored by MERRIN contains $\approx 1.8 \times 10^{15}$ locally monotone BNs, including the gold standard.

Degraded time series generation. We used the workflow in Fig. 1(b) to generate a benchmark of 240 time series sets. First FlexFlux (Marmiesse *et al.*, 2015) was used to generate complete *kinetic-fluxomic-transcriptomic* (KFT) d-rFBA simulation data for the five environmental conditions of the core RMN (see Suppl. Sect. 3.1), each yielding 301 RMSS (initial biomass = $0.1g.L^{-1}$, steps = 300, intervals = $0.01h$). Then, for each complete KFT time series, we generated (i) a *kinetic-fluxomic* (KF) time series by removing the values of the regulated proteins, (ii) a *kinetic-transcriptomic* (KT) time series by discretizing all fluxes to binary values (iii) a *transcriptomic* (T) time series by discretizing all fluxes and metabolite concentrations to binary values. The resulting time series were further compressed by removing redundant time points to emulate biological experiments where only a few selected measurements are made. Finally, for each of the five environmental conditions and each type of data (KFT, KF, KT, T), we generated 60 random time series at different noise rates (0%, 10%, 20%, 30%, 40% and 50%), by randomly deleting time points and increasing or decreasing quantitative values. Altogether we



(a) MERRIN software



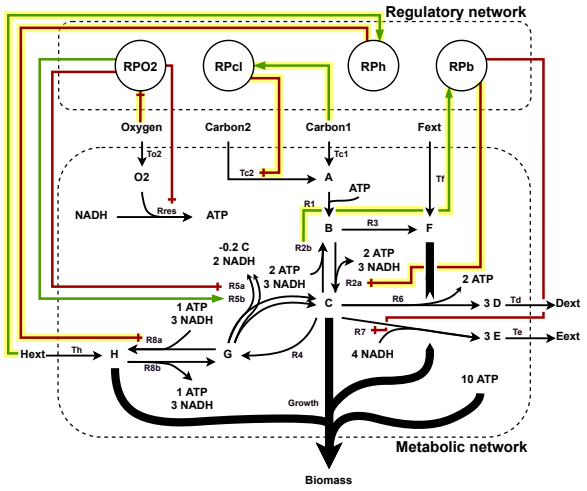
(b) Data generation procedure

Fig. S1. **(a)** Workflow of the **MERRIN software** for metabolic regulation rule inference. **(b) Degraded time series generation procedure**: generation of 240 time series for the RMN of (Covert et al., 2001), with different levels of incompleteness and noise.

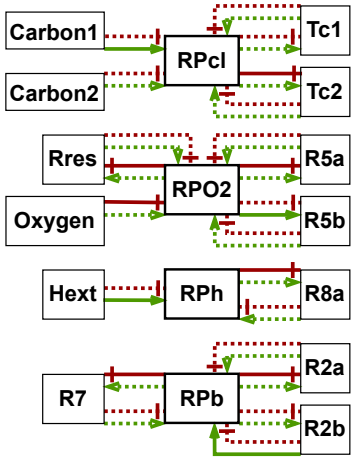
obtained 240 sets of 5 incomplete and/or noisy time series, each including 6 to 18 time points after the compression step.

Inference scores. The quality of MERRIN predictions was evaluated on two different levels. First, we measured the distance between the observed time series, on which the inference was based, and the time series obtained by simulating the inferred model. The distance between two RMSS time series $S = \{s^0, \cdots, s^m\}$ and $\hat{S} = \{\hat{s}^0, \cdots, \hat{s}^m\}$ w.r.t. a set of components A was computed as the *residual sum of squares* (RSS): $RSS_A = \sum_{i=0}^{m} \sum_{a \in A} (s_a^i - \hat{s}_a^i)^2$. We used $\mathrm{RSS}_{\mathcal{P}}$ to measure the accuracy of the prediction of the time series for the four regulatory proteins (RPcl, RPO2, RPh, RPb) and $\mathrm{RSS}_{\mathrm{Ext}}$ to measure the accuracy of the prediction of the time series of the eight external metabolites (Carbon1, Carbon2, Oxygen, Hext, Fext, Dext, Eext, Biomass).

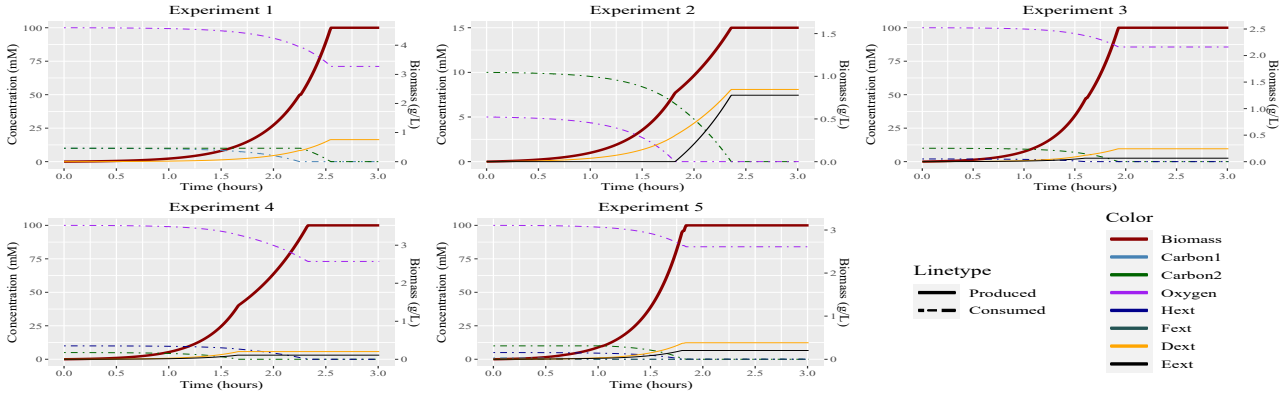
Second, we measured the ability of MERRIN to infer the expected regulations using the recall and precision of the inferred BN. Given BNs



(a) Regulated metabolic network (RMN)



(b) Prior knowledge network (PKN)



(c) Inferred RMN simulations

Fig. S2. **(a)** Regulated metabolic network from (Covert et al., 2001). Lower part is the metabolic network. The nodes are metabolites and the black hyperedges are reactions. Upper part is the regulatory network. The nodes are regulatory proteins. Edges represent the Boolean functions: green edges denote activation, red edges inhibition. Yellow highlighted edges are the inferred regulation from the complete noise-free time series. **(b)** Set of permitted interactions use for the inference. Red edges, solid and dot, are inhibitions. Green edges, solid and dot, are activations. The set of solid edges describes the influence graph of the regulatory network of (a). **(c)** FlexFlux simulations of the inferred RMN (yellow highlighted regulations in (a)) using the experimental conditions of (Covert et al., 2001). These simulations are identical to the simulations of the reference RMN.

f and $\hat{f}$, the *recall* of $G(\hat{f})$ w.r.t. $G(f)$ is the fraction of edges of $G(f)$ in $G(\hat{f})$, *i.e.*, recall $= |G(f) \cap G(\hat{f})|/|G(f)|$, where $|G(f)|$ denotes the number of edges. The *precision* of $G(\hat{f})$ w.r.t. $G(f)$ is the fraction of edges of $G(\hat{f})$ in $G(f)$, *i.e.*, precision $= |G(f) \cap G(\hat{f})|/|G(\hat{f})|$.

3.3 Performance of MERRIN on complete data

MERRIN was first applied to the complete noise-free kinetic-fluxomics-transcriptomics (KFT) time series corresponding to the five different environmental conditions. On this input, MERRIN inferred exactly one smallest regulatory BN in 6.95s. The inferred regulatory rules are shown with yellow highlighted edges in Fig. 2(a). The BN contains seven regulatory rules (for RPO2, RPcl, RPh, RPb, Tc2, R2a and R8a) of the gold standard, three of which regulate reaction activity. It has a *precision* of 1, meaning that all seven regulatory rules are in the gold standard; and a *recall* of 0.64, because four of the regulatory rules of the gold standard have not been retrieved (rules for R5a, R5b, R7 and Rres). Both *RSS*s are equal to 0: although the recall is not 1, the d-rFBA simulations of the five experiments with the inferred regulatory BN (Fig. 2(c)) match exactly the complete noise-free time series. The unrecovered regulatory rules of the gold standard are not necessary to explain the observed time series.

This is consistent with the discussion in (Covert *et al.*, 2001) that the regulation of *Rres* is not necessary for the optimal solution. Biologically, this regulation is only present to ensure that unnecessary respiratory enzymes decay in an anaerobic environment. However, since enzyme amounts are not explicitly represented in the d-rFBA framework, the time series do not reflect this biological behavior, hampering the inference of the regulation. Similarly, R5a and R5b were introduced in the RMN to model that aerobic and anaerobic carbon synthesis is catalyzed by different enzymes. However, these enzymes are not included in the model and both reactions are strictly equivalent. It is therefore not surprising that MERRIN cannot infer the regulation stating which of the two reactions should be selected. Finally, the missing regulation of R7 in the inferred RMN is explained by the fact that R7 cannot be activated in d-rFBA simulations optimizing growth because its activation would consume carbon and energy, leading to a decrease in biomass synthesis. Therefore, regulating $R7$ is not necessary to explain its activity in the simulations.

3.4 Impact of data incompleteness and noise

Range of application of MERRIN. When considering higher degradation rates (40% and 50%), 9 of the 60 test instances reached the time limit of

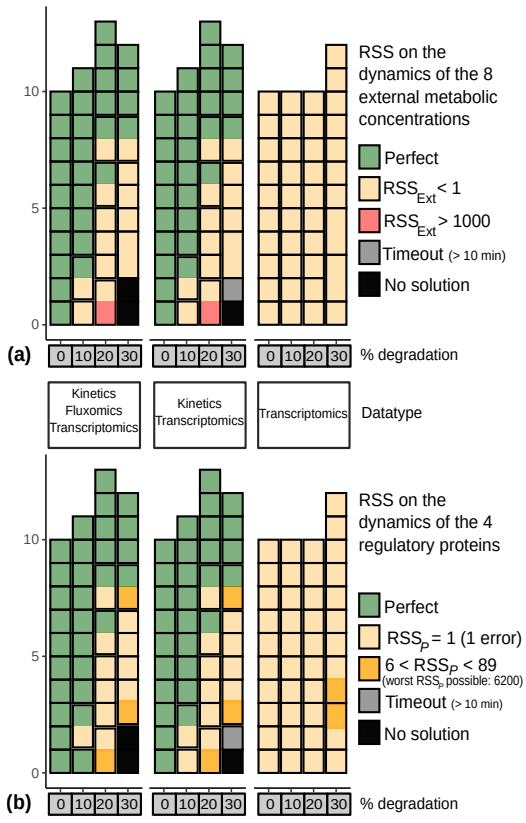


Fig. S3. RSS depending on data type and degradation level. Each vertical bar corresponds to the results of MERRIN on the 10 instances associated with a considered data type (KFT, FT, T) and degradation type (0%, 10%, 20% and 30%). Each square corresponds to one solution and its color to RSS ranges (see legend). A black edge separates the MERRIN results on the different instances.

600s (see Suppl. Sec. 3.2.1). The number of BNs also increased drastically at 50% degradation, as well as the RSS scores, suggesting that the degradation rate of 30% is the limit for the MERRIN approach. As shown in Suppl. Sect. 3.2.2, we also tested the case of kinetic-fluxomics instances. Such instances do not contain any information on the four regulatory protein states, making it difficult to infer regulatory rules between proteins and reactions. As expected, MERRIN is not able to correctly determine the regulatory rules controlling them. This leads to time-consuming enumeration of a very large number of BNs, all compatible with the observed time series, but considering all the possible regulatory protein states. Based on these results, we suggest to use MERRIN only on kinetics and transcriptomics real data sets. According to the design of MERRIN, proteomics data can be viewed as alternative to transcriptomics data if they are available. Therefore, in the following, we focus only on the data types KFT (kinetic-fluxomics-transcriptomics), KT (kinetics-transcriptomics) and T (transcriptomics) with a degradation rate between 0 and 30%, which represents 120 instances.

Number of models inferred by MERRIN. Fig. S3 shows the number of subset-minimal models inferred by MERRIN in the given time limit for the 120 tested instances . When a solution was reached in the time limit, MERRIN inferred at most two subset-minimal models. In total, 134 BNs were inferred from the 120 instances. Among these 134 BNs, there were only 15 different BNs, see Fig. 2 in the Suppl. Sect. 3.2.2. For each of these models, we computed the precision and recall (see Sect.3.2) with respect to the gold standard (see Suppl. Sect. 3.2.3, Fig. 3). For 110 instances out of 120, the precision is equal to 1, meaning that all the regulatory rules inferred in these BNs are present in the gold standard. The maximum recall is equal to 0.64, while the minimum recall is 0.55.

Performance. Among the 120 instances of our benchmark, only one has reached the time limit (grey square in Fig. S3). For this instance, we do not have any information whether or not there is a solution. In 3 out of the 120 instances (Fig. S3), MERRIN reported that no BN satisfied the constraints. This happens only at 30% noise rate. For the 116 other instances, the average inference time was 25.975s.

Simulation scores. For each of the 134 BNs inferred, we compared the associated d-rFBA time series of external metabolites and regulatory proteins to the ones of the gold standard using the $\mathrm{RSS}_{\mathrm{Ext}}$ score (Fig. S3(a)) and the $\mathrm{RSS}_{\mathcal{P}}$ score (Fig. S3(b)). In Fig. S3, green squares correspond to cases where MERRIN inferred a unique BN whose associated RMN has exactly the same r-dFBA simulations as the gold standard ($\mathrm{RSS}_{\mathrm{Ext}} = 0$ (Fig. S3(a)) and $\mathrm{RSS}_{\mathcal{P}} = 0$ (Fig. S3(b))). Interestingly, the same BN was inferred for each green square, and this BN is the same as the one obtained on complete data (Fig. 2(a)) Yellow squares of Fig. S3 stand for BNs reproducing the gold standard RMN simulations with a very small error. These errors are due to missing regulatory rules. For example, all the BNs with $\mathrm{RSS}_{\mathrm{Ext}} < 1$ and $\mathrm{RSS}_{\mathcal{P}} = 1$ are BNs for which the regulatory rule of reaction R2a has not been inferred. Red squares correspond to the worst possible $\mathrm{RSS}_{\mathrm{Ext}}$ ($> 1\,000$), equivalent to cases in which no regulatory rules were inferred. This happens twice among the 120 experiments.

Impact of degradation rate. A vertical bar of 10 green squares in Fig. S3 means that MERRIN inferred, for each of the 10 test instances, a unique BN that perfectly matches the gold standard. This occurred only for KT and KFT instances with no degradation in the input time series. $\mathrm{RSS}_{\mathrm{Ext}}$ and $\mathrm{RSS}_{\mathcal{P}}$ increased with the degradation rate, as one should expect. However, most of the RSS scores are very small, emphasizing that the inferred BNs can almost perfectly reproduce the gold standard when the degradation rates is less than 30%.

Impact of the type of data. The results are identical for the complete (KFT) and the kinetic-transcriptomics (KT) instances (except one KP at 30%, which reached the time limit of 600s). This could be expected since MERRIN reasons over binarized fluxomics data, which once binarized are identical to the qualitative information provided by transcriptomics data. In addition, the inferred BNs from the KFT and KT time series reproduce the gold standard with good precision most of the time, except in two cases (red squares).

For transcriptomics (T) time series instances, our results show that no inferred BN was able to perfectly reproduce the gold standard. However, for each inferred BN both $\mathrm{RSS}_{\mathrm{Ext}}$ and $\mathrm{RSS}_{\mathcal{P}}$ are small: $\mathrm{RSS}_{\mathcal{P}} \leq 1$ for all, except for two instances, and $\mathrm{RSS}_{\mathrm{Ext}} < 1$. This suggests that without information on external metabolite concentrations, it is harder for MERRIN to explain if the observed RMSS is due to some regulations or to a specific combination of external metabolite concentrations. In this case, regulatory rules, such as the rule controlling the reaction R2a, are missed.

4 Discussion and conclusion

We introduced MERRIN, a novel approach to infer rules for metabolic regulation in changing environments. MERRIN is based on the d-rFBA framework, which combines discrete simulations of Boolean networks, modeling the activity of regulatory proteins, with the prediction of metabolic response, based on linear programming.

Advantages of using constraint propagators. A characteristic of the inference problem is that the set of BNs verifying both combinatorial and linear constraints is small compared to the set of BNs verifying only the combinatorial constraints. To address this issue, our resolution implements a Satisfiability Modulo Theory (SMT) approach with a dedicated algorithm for combining Boolean satisfiability with linear programming: we designed a constraint propagation strategy on top of the Answer Set Programming solver Clingo by exploiting a monotonicity property of the optimization objective in RMNs. This strategy reduced substantially the number of candidate solutions to be validated, by generalizing counterexamples satisfying the combinatorial constraints but not the linear ones encountered during the search.

Possible strategies to infer all regulatory rules. MERRIN infers regulations only when they improve the fitting between observations and simulations, which depends on the underlying optimality principle (here optimizing growth). Since the presence of some regulations from the gold standard does not affect the fitting, it is not possible for MERRIN to infer them. Inferring more regulations would require to introduce enzyme amounts and their synthesis. Methods such as r-deFBA (Liu and Bockmayr, 2020), should allow solving this issue.

Impact of the synchronous simulation assumption. The d-rFBA framework as defined in (Covert *et al.*, 2001; Marmiesse *et al.*, 2015) uses synchronous simulation of BNs (the state of all regulatory proteins is updated simultaneously). While our implementation allows considering asynchronous simulation, this results in a less constrained model. Indeed, the fact that a regulatory protein has the same state in two consecutive steady states could be explained either with the application of a regulatory rule, or by the absence of an update. Therefore, considering asynchronous updates would probably require considering further time constraints in order to match the experimental observations.

Use of synthetic data to validate network inference. The validation of methods related to the inference of regulatory rules can be misleading since there is no reference multi-layer data set or reference RMN allowing large-scale validations. As discussed in (Covert *et al.*, 2001) and confirmed in (Thuillier *et al.*, 2021), even in the most complete (small-scale) gold standard RMN introduced in (Covert *et al.*, 2001), some regulatory rules introduced according to literature-based knowledge have no impact on the RMN simulation. To address this issue and to test our approach, we used a benchmark strategy consisting in generating several types of data from the simulations of a gold standard. This allowed testing the robustness of the MERRIN approach in different *scenarios* of data types (combinations of kinetics, fluxomics and transcriptomics data) and noise (up to 50% noise introduced in the data). We argue that such a benchmark strategy could be used in a similar way to test the robustness of any other dynamical network inference method when only few reference data are available.

Impact of data types and quality. According to our results, the performance of MERRIN on kinetic and transcriptomics data is similar to complete data (kinetic, fluxomics and transcriptomics). This suggests that inferring regulatory rules of metabolic networks actually would not require fluxomics data, which are most probably the hardest data to obtain experimentally. In this direction, a perspective to extend the MERRIN approach would be to identify the best experimental designs to discriminate the models associated with the PKN. In addition, MERRIN seems to be sensitive to noise only for single fluxomics data. In all other cases, up to 30% noise in the data has few impact of the MERRIN performance.

Scalability. The computation times in this study are encouraging for inferring regulations in larger networks. Handling linear constraints reduces to FBA, which can be done efficiently on genome-scale networks. However, this has to be done many times during combinatorial search. Thus, for inferring large-scale regulated metabolic networks improved constraint propagation techniques may become necessary to further prune the combinatorial search space.

Funding

Work of LP is supported by the French Agence Nationale pour la Recherche (ANR), grant number ANR-20-CE45-0001. Work of LC and CB is supported by the French Laboratory of Excellence project "TULIP" (grant number ANR- 10- LABX- 41; ANR- 11- IDEX- 0002- 02).

References

Baral, C. (2003). *Knowledge Representation, Reasoning and Declarative Problem Solving*. Cambridge University Press, New York, NY, USA.

Barrett, C.*et al.* (2018). *Satisfiability Modulo Theories*, pages 305–343. Springer International Publishing, Cham.

Bernot, G.*et al.* (2004). Application of formal methods to biological regulatory networks: extending thomas' asynchronous logical approach with temporal logic. *J of Theo Biol*, **229**(3), 339–347.

Chaves, M.*et al.* (2010). Comparing boolean and piecewise affine differential models for genetic networks. *Acta Biotheor*, **58**(2-3), 217–232.

Chevalier, S.*et al.* (2019). Synthesis of boolean networks from biological dynamical constraints using answer-set programming. In *ICTAI*. IEEE.

Covert, M.W.*et al.* (2001). Regulation of gene expression in flux balance models of metabolism. *J of Theo Biol*, **213**(1), 73–88.

de Jong, H. (2002). Modeling and simulation of genetic regulatory systems: A literature review. *J of Comp Biol*, **9**, 67–103.

Feist, A.M.*et al.* (2010). The biomass objective function. *Curr Opin Microbiol*, **13**(3), 344–349.

Forrest, J.*et al.* (2022). coin-or/cbc: Release releases/2.10.7.

Frioux, C.*et al.* (2019). Hybrid metabolic network completion. *Theory and Practice of Logic Programming*, **19**(1), 83–108.

Gebser, M.*et al.* (2012). *Answer Set Solving in Practice*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan and Claypool Publishers.

Gebser, M.*et al.* (2017). Multi-shot ASP solving with clingo. *CoRR*, **abs/1705.09811**.

Goelzer, A.*et al.* (2015). Quantitative prediction of genome-wide resource allocation in bacteria. *Metabolic Engineering*, **32**, 232–243.

Janhunen, T.*et al.* (2017). Clingo goes linear constraints over reals and integers. *Theory and Practice of Logic Programming*, **17**(5-6), 872–888.

Liu, L.*et al.* (2020). Regulatory dynamic enzyme-cost flux balance analysis: A unifying framework for constraint-based modeling. *J of Theo Biol*, **501**, 110317.

Mahadevan, R.*et al.* (2002). Dynamic flux balance analysis of diauxic growth in escherichia coli. *Biophysical Journal*, **83**(3), 1331–1340.

Marmiesse, L.*et al.* (2015). FlexFlux: combining metabolic flux and regulatory network analyses. *BMC Systems Biology*, **9**(1).

Monod, J. (1942). Recherches sur la croissance des cultures bacteriennes. *Ann. Inst. Pasteur*, **69**, 179.

Orth, J.D.*et al.* (2010). What is flux balance analysis? *Nat Biotechnol*, **28**(3), 245–248.

Razzaq, M.*et al.* (2018). Computational discovery of dynamic cell line specific boolean networks from multiplex time-course data. *PLOS Comp Biol*, **14**(10), e1006538.

Saez-Rodriguez, J.*et al.* (2009). Discrete logic modelling as a means to link protein signalling networks with functional analysis of mammalian signal transduction. *Mol Syst Biol*, **5**(1), 331.

Thuillier, K.*et al.* (2021). Learning boolean controls in regulated metabolic networks: A case-study. In *CMSB*, volume 12881 of *LNCS*, pages 159–180. Springer.

Tournier, L.*et al.* (2017). Optimal resource allocation enables mathematical exploration of microbial metabolic configurations. *J. Math. Biol.*, **75**(6-7), 1349–1380.

Tsiantis, N.*et al.* (2018). Optimality and identification of dynamic models in systems biology: an inverse optimal control framework. *Bioinf*, **34**(14), 2433–2440.

Varma, A.*et al.* (1994). Stoichiometric flux balance models quantitatively predict growth and metabolic by-product secretion in wild-type Escherichia coli W3110. *Appl Environ Microbiol*, **60**(10), 3724–3731.

Videla, S.*et al.* (2017). caspo: a toolbox for automated reasoning on the response of logical signaling networks families. *Bioinf*, page btw738.

Bioinformatics, 38, 2022, ii127–ii133
https://doi.org/10.1093/bioinformatics/btac479
Proceedings of ECCB2022

OXFORD

Systems Biology

Supplementary Materials — MERRIN: MEtabolic Regulation Rule INference from time series data

Kerian Thuillier [1], Caroline Baroukh [2], Alexander Bockmayr [3], Ludovic Cottret [2], Loïc Paulevé [4], Anne Siegel [1]*

[1]Univ Rennes, Inria, CNRS, IRISA, F-35000 Rennes, France
[2]LIPME, INRAE, CNRS, Université de Toulouse, Castanet-Tolosan, France
[3]*Freie Universität Berlin, Institute of Mathematics, D-14195 Berlin, Germany
[4]*Univ. Bordeaux, Bordeaux INP, CNRS, LaBRI, UMR5800, F-33400 Talence, France

Abstract

This is the supplementary file of: *MERRIN: MEtabolic Regulation Rule INference from time series data*.

1 Notations

$|X|$. The cardinality of a finite set X is denoted by $|X|$.

x_I. Given a vector $x \in D^n$ and a set of indices $I \subseteq \{1, \cdots, n\}$, x_I denotes the vector of dimension $|I|$ equal to $(x_i)_{i \in I}$.

$\mathbb{B}$. The Boolean domain is denoted by $\mathbb{B} = \{0, 1\}$.

$\beta(s)$. Given a non-negative real vector $s \in \mathbb{R}^n_{\geq 0}$, we denote by $\beta(s) \in \mathbb{B}^n$ its *binarization*, *i.e.* $\forall i \in \{1, \ldots, n\}, \beta(s)_i = 1$, if $s_i > 0$, and $\beta(s)_i = 0$, if $s_i = 0$.

2 Boolean over-approximation of RMSS

This section is a complement to Sect. 2.1.3 and details the Boolean relaxation of the Eq.(2.c).

Boolean over-approximation of regulatory-metabolic steady state ($\mathbb{B}$-RMSS) of a RMN $(\mathcal{N}, \mathrm{Inp}, \mathcal{P}, f)$ can be defined as a triplet $(\overline{v}, \overline{c}, \overline{x}) \in \mathbb{B}^{|\mathcal{R}|} \times \mathbb{B}^{|\mathrm{Ext}|} \times \mathbb{B}^{|\mathrm{Inp}|+|\mathcal{P}|+|\mathcal{R}|}$ (Thuillier *et al.*, 2021) associating binary reaction states $\overline{v}$, external metabolite availabilities $\overline{c}$, and a regulatory state $\overline{x}$. The binary reaction states $\overline{v}$ must satisfy a relaxed form of Eqs.(1):

$$(1.a_{\text{relaxed}})\ \forall m \in \mathrm{Int}, \bigvee_{r \in \mathcal{R} S_{mr} > 0} \overline{v}_r \iff \bigvee_{r \in R, S_{r,m} < 0} \overline{v}_r$$

$$(1.b_{\text{relaxed}})\ \forall r \in \mathcal{R},\ \overline{x}_r = 0 \implies \overline{v}_r = 0$$

$$(1.c_{\text{relaxed}})\ \forall m \in \mathrm{Inp},\ \forall r \in \mathcal{R},\ S_{mr} < 0 \implies \overline{v_r} \leq \overline{c}_m$$

Let us denote by $\overline{\mathbb{S}}$ the set of all the $\mathbb{B}$-RMSS of the RMN $(\mathcal{N}, \mathrm{Inp}, \mathcal{P}, f)$ (satisfying the relaxed equations Eqs.(1_{relaxed}). Eq.(2.c) can be relaxed by considering $\mathbb{B}$-RMSS instead of RMSS, thus:

$$(2.c_{\text{relaxed}}) \qquad (v', c', x') \in \overline{\mathbb{S}}$$

It must be noted that the set of binarised RMSS is included in $\mathbb{S}$, *i.e.* $\forall (v, c, x) \in \mathbb{S},\ (\beta(v), \beta(c), x) \in \overline{\mathbb{S}}$. The converse is not true.

3 Results

3.1 Experiment conditions of Covert *et al.* (2001)

This section is a complement to Sect. 3.2 and details the initial states of the 5 experiment conditions described in Covert et al. *(2001).*

The 5 experiment conditions of Covert *et al.* (2001), used to generate our experimental time series, are shown in Tab. S1. Each experiment is based on a different set of initial input metabolite concentrations c and the regulatory state x is initialized such that: (i) $\forall r \in \mathcal{R}, x_r = 0$, (ii) $\forall i \in \mathrm{Inp}, x_i = \beta(c_i)$, (iii) for each regulatory protein we apply the associated regulatory rule: $x_{\text{RPcl}} = \beta(c_{\text{Carbon1}})$, $x_{\text{RPO2}} = \beta(c_{\text{Oxygen}})$, $x_{\text{RPb}} = 0$ and $x_{\text{RPh}} = \beta(c_{\text{Hext}})$.

	Input metabolite concentration (mmol.L^{-1})					Regulatory protein state			
Experiment	c_{Carbon1}	c_{Carbon2}	c_{Oxygen}	c_{Fext}	c_{Hext}	x_{RPcl}	x_{RPO2}	x_{RPb}	x_{RPh}
1	10	10	100	0	0	1	0	0	0
2	0	10	5	0	0	0	0	0	0
3	0	10	100	0	2	0	0	0	1
4	0	5	100	0	10	0	0	0	1
5	1	10	100	0.1	5	1	0	0	1

Table S1. Experiment conditions used to generate the 5 simulations of (Covert et al., 2001).

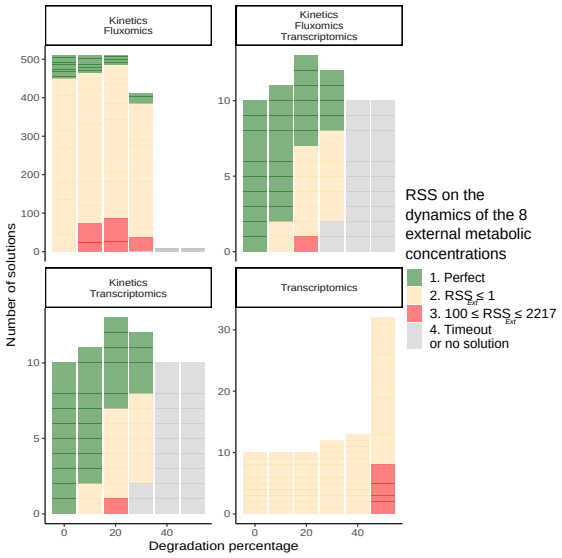
3.2 Inferring from non-complete noisy time series

This section is a complement to Sect. 3.4.

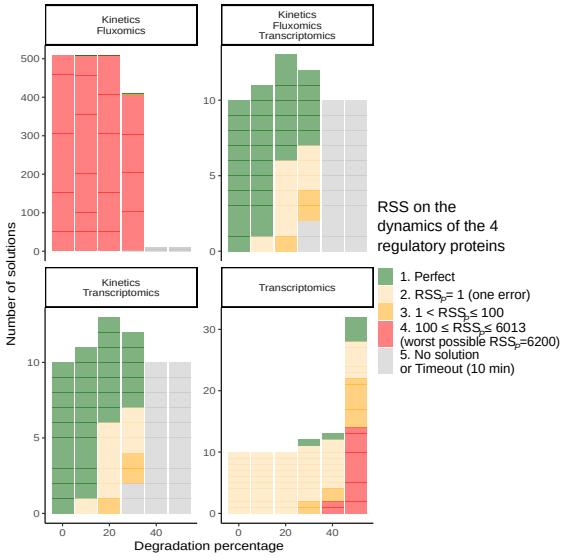
3.2.1 Comparing the d-rFBA simulations

RSS scores (RSS_{Ext} and $RSS_{\mathcal{P}}$) of the regulatory inferred on the 240 instances.

For each regulatory BN inferred for the 240 instances, we compared the associated d-rFBA time series of external metabolites and regulatory proteins to the ones of the gold standard model using the $\mathrm{RSS}_{\mathrm{Ext}}$ score (Fig. S1(a)) and the $\mathrm{RSS}_{\mathcal{P}}$ score (Fig. S1(b)).



(a) $\mathrm{RSS}_{\mathrm{Ext}}$ on the dynamics of the 8 external metabolic concentrations



(b) $\mathrm{RSS}_{\mathcal{P}}$ on the dynamics of the 4 regulatory proteins

Fig. S1. RSS depending on datatype and degradation level. Each vertical bar corresponds to the results of MERRIN on the 10 instances associated with a considered datatype (KF, KFT, KT, T) and degradation level (0%, 10%, 20%, 30%, 40%, 50%). The different colors represents the score range ((a) $\mathrm{RSS}_{\mathrm{Ext}}$ and (b) $\mathrm{RSS}_{\mathcal{P}}$) of the solution (see legend).

Kinetic-fluxomics instances. Kinetic-fluxomics (KF) instances do not contain any information on the four regulatory protein states. As expected, MERRIN is not able to determine the regulatory protein states (Fig. S1(b)) from this datatype leading to the enumeration of a huge number of compatible BNs. Here, we have restricted the number of solutions to enumerate to 51, this limit was reached for each (KF) instance that admits a solution to the inference problem. Thus, we do not recommend to use MERRIN on KF instances.

Impact of the degradation rate. Our results show that MERRIN could not infer any regulatory BNs on the datatype KFT, KF, and KT with a degradation rate strictly greater than 30%. For T instance, the number of inferred BNs increased significantly at 40% and 50% of degradation. Moreover, at high degradation level, both $\mathrm{RSS}_{\mathcal{P}}$ and $\mathrm{RSS}_{\mathrm{Ext}}$ decrease drastically: a huge part of the BNs inferred for T instances have an $\mathrm{RSS}_{\mathcal{P}}$ and an $\mathrm{RSS}_{\mathrm{Ext}}$ greater than 100. Thus, we do not recommend to use MERRIN on instances having a degradation level higher than 30%.

3.2.2 Inferred regulatory BNs

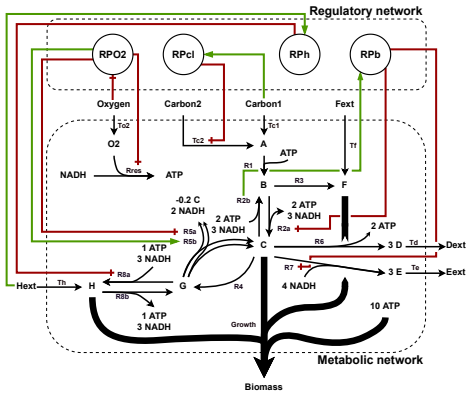
Enumerations of the 15 different regulatory BNs inferred from 120 instances.

In this section, we focus on the results obtained on 120 different time-series instances: complete (KFT), kinetic-transcriptomics (KT), and transcriptomics (T) instances with a noise ranging from 0% to 30% (Sect. 3.4).

Let us consider the metabolic network $\mathcal{N}$, the set of inputs metabolites Inp and the set of regulatory proteins $\mathcal{P}$ given as input to MERRIN. There are 15 different regulatory BNs that have been inferred on the 120 instances, for each inferred regulatory BN f, the RMN $(\mathcal{N}, \mathrm{Inp}, \mathcal{P}, f)$ is shown in Fig. S2 with their respective scores: precision, recall, $\mathrm{RSS}_{\mathcal{P}}$, $\mathrm{RSS}_{\mathrm{Ext}}$.

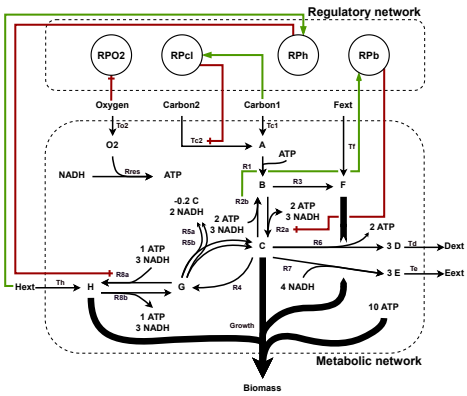
Best result. The regulatory BN of Fig. S2(b) is the one inferred on the complete (KFT) datatype with 0% of degradation. It allows exactly reproducing the 5 d-rFBA simulations of Covert *et al.* (2001) used to generate the input time series. This BN has been inferred on 58 of the 120 instances and only on the datatype KFT and KT.

Worst result. Among the 15 regulatory BN, the regulatory BN of Fig. S2(p) has the worst RSSs scores: $\mathrm{RSS}_{\mathcal{P}} = 89$ and $\mathrm{RSS}_{\mathrm{Ext}} = 1194.07$. These scores are due to the absence of regulation controlling the reaction *Tc2* which inhibits the consumption of *Carbon2* if some *Carbon1* is available.



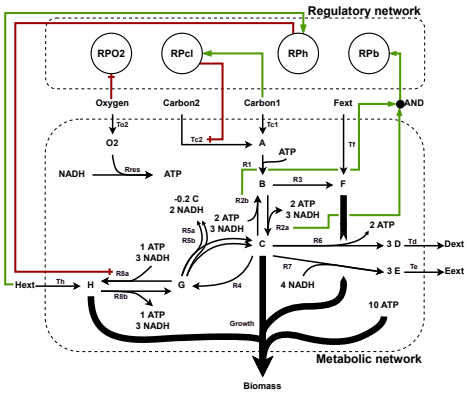
Precision: 1; Recall: 1; $\mathrm{RSS}_{\mathcal{P}}$: 0; $\mathrm{RSS}_{\mathrm{Ext}}$: 0

(a) **Gold standard RMN**



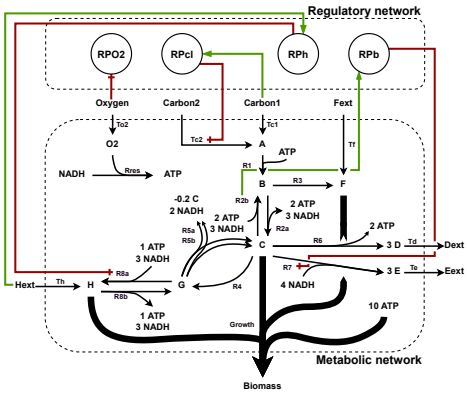
Precision: 1; Recall: 0.64; $\mathrm{RSS}_{\mathcal{P}}$: 0; $\mathrm{RSS}_{\mathrm{Ext}}$: 0

(b)



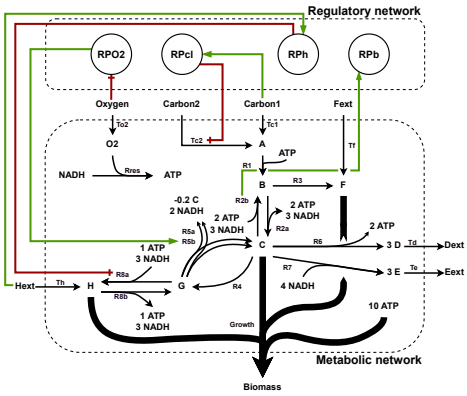
Precision: 0.86; Recall: 0.55; $\mathrm{RSS}_{\mathcal{P}}$: 68; $\mathrm{RSS}_{\mathrm{Ext}}$: 0.18

(c)



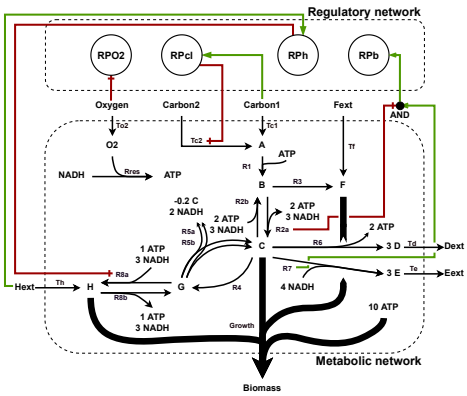
Precision: 1; Recall: 0.64; $\mathrm{RSS}_{\mathcal{P}}$: 1; $\mathrm{RSS}_{\mathrm{Ext}}$: 0.12

(d)



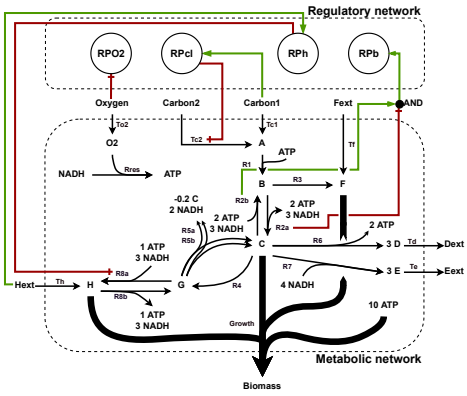
Precision: 1; Recall: 0.64; $\mathrm{RSS}_{\mathcal{P}}$: 1; $\mathrm{RSS}_{\mathrm{Ext}}$: 0.18

(e)



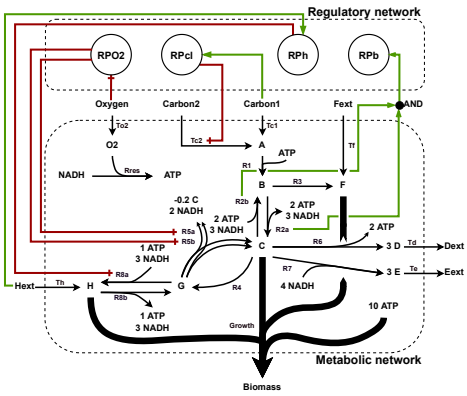
Precision: 0.71; Recall: 0.45; $\mathrm{RSS}_{\mathcal{P}}$: 68; $\mathrm{RSS}_{\mathrm{Ext}}$: 0.18

(f)



Precision: 0.86; Recall: 0.55; $\mathrm{RSS}_{\mathcal{P}}$: 1; $\mathrm{RSS}_{\mathrm{Ext}}$: 0.18

(g)

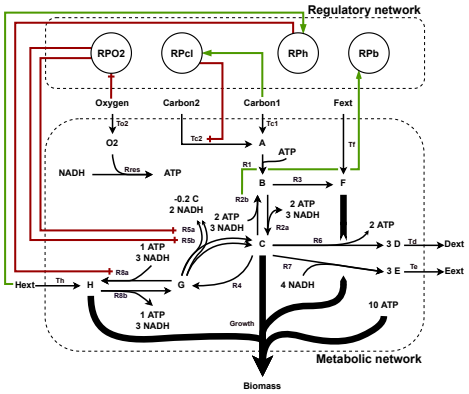


Precision: 0.78; Recall: 0.64; $\mathrm{RSS}_{\mathcal{P}}$: 68; $\mathrm{RSS}_{\mathrm{Ext}}$: 0.18

(h)

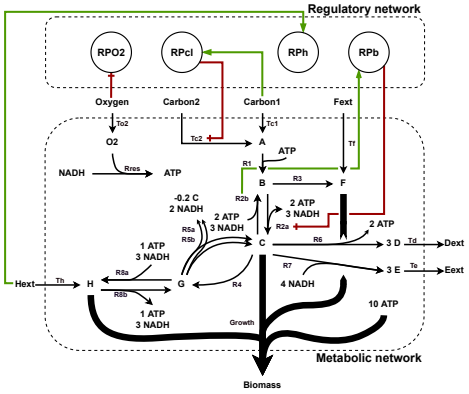
Precision: 1; Recall: 0.64; $\mathrm{RSS}_{\mathcal{P}}$: 1; $\mathrm{RSS}_{\mathrm{Ext}}$: 0.18

(i)



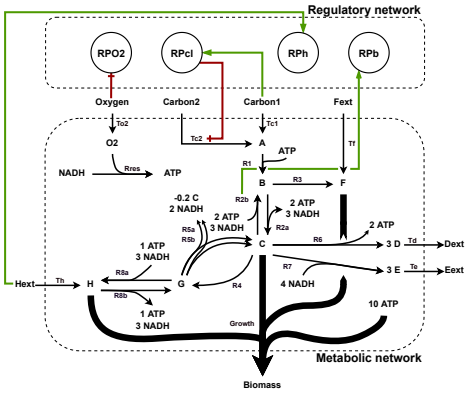
Precision: 0.88; Recall: 0.64; $\text{RSS}_{\mathcal{P}}$: 1; RSS_{Ext}: 0.18

(j)



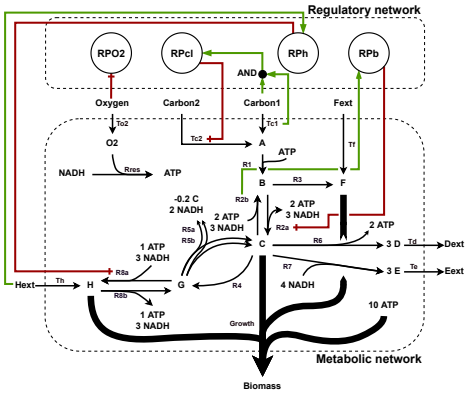
Precision: 1; Recall: 0.55; $\text{RSS}_{\mathcal{P}}$: 0; RSS_{Ext}: 0.41

(k)



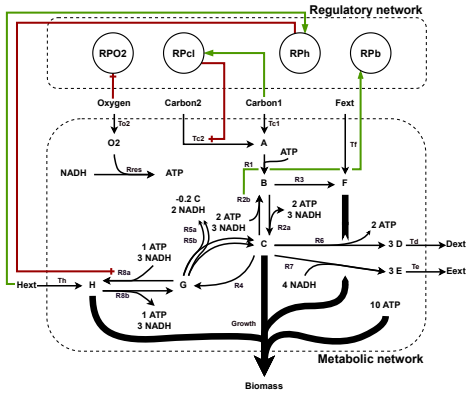
Precision: 1; Recall: 0.45; $\text{RSS}_{\mathcal{P}}$: 1; RSS_{Ext}: 0.5

(l)



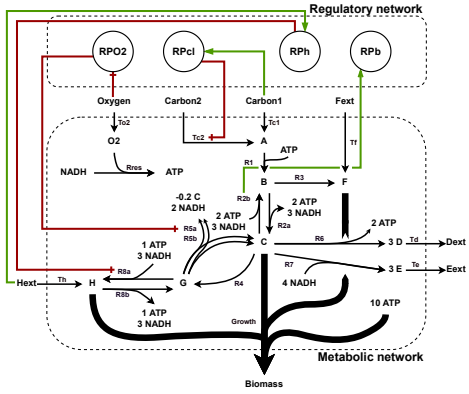
Precision: 0.75; Recall: 0.55; $\text{RSS}_{\mathcal{P}}$: 6; RSS_{Ext}: 0.94

(m)



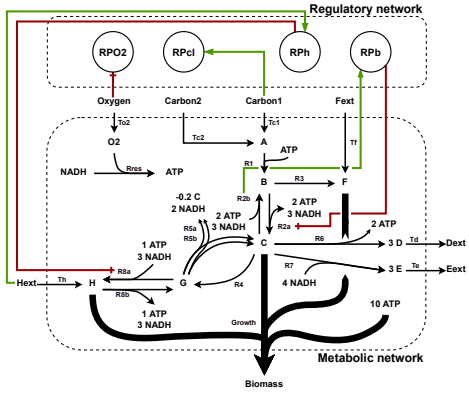
Precision: 1; Recall: 0.55; $\text{RSS}_{\mathcal{P}}$: 1; RSS_{Ext}: 0.18

(n)



Precision: 1; Recall: 0.64; $\text{RSS}_{\mathcal{P}}$: 1; RSS_{Ext}: 0.18

(o)



Precision: 1; Recall: 0.55; $\text{RSS}_{\mathcal{P}}$: 89; RSS_{Ext}: 1194.07

(p)

Fig. S2. (a) Regulatory BN of the gold standard model (Covert et al., 2001). (b)-(p) Set of 15 regulatory BNs inferred from the 120 instances representing kinetic-fluxes-transcriptomics (KFT), kinetic-transcriptomics (KT), and transcriptomics (T) observations with a noise ranging from 0% to 30%.

3.2.3 Comparisons with the gold standard regulatory BN

Recall and precision scores of the inferred BNs.

Let us focus on the 120 instances (datatype KFT, KT, and T with a degradation level between 0% and 30%). For each inferred BN, we computed the recall and the precision according to the gold standard regulatory BN. Fig. S3 represents the worst recall and the worst precision of

each one of the 120 instances depending of the datatype and the degradation

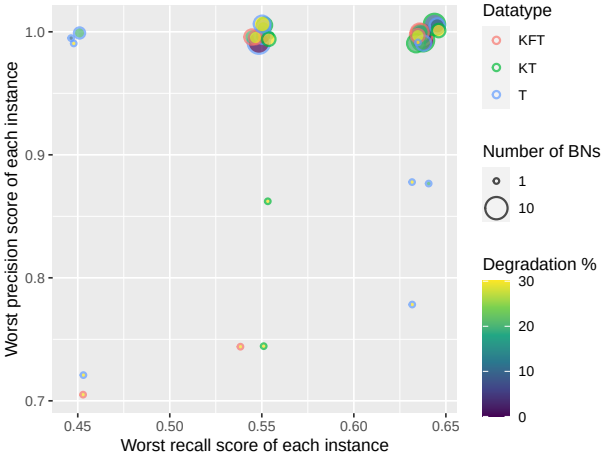


Fig. S3. Worst recall and precision depending of data type and degradation level. For each instance of the considered datatype (KFT, KT, T) and degradation level (0%, 10%, 20%, 30%), only the worst recall and worst precision are considered. Each circle corresponds to a set of instances of identical datatype and degradation levels having the same worst recall and worst precision.

level. Our results show that, except for 8 instances, MERRIN inferred BNs having a precision of 1 and a recall between 0.45 and 0.64, meaning that at least 50% of the edge of the influence graph of the gold standard are correctly retrieved.

The degradation level seems to have the greatest impact on the precision score: all, except one, instances with a worst precision lower than 1 have a degradation level of 30%. For the recall, it appears that it is the datatype that has the bigger impact: T instances have a smaller recall than the other KFT and KT instances. This last result can be easily explained by the fact that T instances do not have any information on the input metabolite concentrations, thus it is harder to define if an observed RMSS is due to a specific concentration of input metabolites or to some regulatory states.

References

Covert, M.W.*et al.* (2001). Regulation of gene expression in flux balance models of metabolism. *J of Theo Biol*, **213**(1), 73–88.

Thuillier, K.*et al.* (2021). Learning boolean controls in regulated metabolic networks: A case-study. In *CMSB*, volume 12881 of *LNCS*, pages 159–180. Springer.

V A Generic Solving Framework for Optimization Modulo Quantified Linear Arithmetic Problems

In this chapter, we introduce a generic solving framework for optimization problems under logic and quantified linear constraints, and its implementation *MerrinASP*. This framework is a generalization of the hybrid solving framework developed for solving the flux-based inference problem in Chapter IV. The results of this chapter have been presented at the *Association for the Advancement of Artificial Intelligence* conference (AAAI) of 2024 and the associated paper published in *Proceedings of the 38th AAAI Conference on Artificial Intelligence* (Thuillier et al., 2024).

To sum up

The inference problem is an example of an optimization problem under logic and quantified linear constraints (OPT+qLP). This chapter introduces a novel generic framework to solve OPT+qLP problems, and its implementation *MerrinASP*. It generalizes the hybrid solving method used for the flux-based inference problem of Chapter IV to solve a broader class of optimization problems. It relies on the so-called *Counter-Example Guided Abstract Refinement* (CEGAR) methods and monotone properties over linear problem structures to learn new constraints and filter spurious candidate solutions directly during the solving process. Our implementation of this framework, *MerrinASP*, has shown to be 10 times more efficient in solving the inference problem than other ASP-based hybrid solvers. In particular, it is now possible to solve the inference problem on medium-scale regulated metabolic networks.

In this chapter

1 Problem Statement

OPT+qLP problems. In Chapter IV, we introduced a dedicated hybrid solving framework for the flux-based formulation of the inference problem. This specific formulation of the inference problem is part of a broader class of hybrid optimization problems: optimization problems under logical constraints and quantified linear constraints (OPT+qLP).

For the flux-based inference problem, the phenotype compatibility is expressed as a constraint over the maximal growth allowed by a substrate state w and a regulatory state x. Let $\mathrm{rMSS}(\mathcal{N}, w, x)$ denote the set of metabolic states of the metabolic network $\mathcal{N}$ that are compatible with w and x. Given the observed growth $\hat{v}_{\text{growth}}$ and a noise rate parameter $0 \leq \epsilon < 1$, we recall the definition of the phenotype compatibility for the flux-based inference in Eqs. V.1.

$$\frac{\hat{v}_{\text{growth}}}{1+\epsilon} \leq v_{\text{growth}} \qquad \text{(V.1a)} \qquad \max_{v' \in \mathrm{rMSS}(w,x)} v'_{\text{growth}} \leq \frac{\hat{v}_{\text{growth}}}{1-\epsilon} \qquad \text{(V.1b)}$$

These equations can be rewritten as equivalent formulas using universal quantifiers instead of the maximization operator (Eqs. V.2). Indeed, if the maximal growth satisfies the upper bound in equations V.1b, then all regulated metabolic steady-

states (RMSSs) have growths that satisfy the upper bound (V.2b).

$$\exists\, v' \in \mathrm{rMSS}(w,x),\ \frac{\hat{v}_{\mathrm{Growth}}}{1+\epsilon} \leq v'_{\mathrm{Growth}} \tag{V.2a}$$

$$\forall\, v' \in \mathrm{rMSS}(w,x),\ v'_{\mathrm{Growth}} \leq \frac{\hat{v}_{\mathrm{Growth}}}{1-\epsilon} \tag{V.2b}$$

OPT+qLP abstraction. Based on this reformulation of phenotype compatibility, we define a new formulation of the inference problem equivalent to the flux-based formulation: the OPT+qLP formulation of the inference problem.

OPT+qLP formulation of the inference problem

Input:

1: a metabolic network $\mathcal{N} = (\mathcal{M}_{\mathrm{ext}} \cup \mathcal{M}_{\mathrm{int}}, \mathcal{R}, s)$ with an objective reaction 'growth';
2: a set of regulatory proteins $\mathcal{P}$;
3: a set of observed time series $\{\mathcal{T}_o^1, \cdots, \mathcal{T}_o^q\}$, $q \geq 1$;
4: a prior knowledge network $\mathcal{G}$ of dimension $n = |\mathcal{P}| + |\mathcal{M}_{\mathrm{ext}}| + |\mathcal{R}|$;
5: a maximum distance $K_{\max} \in \mathbb{N}$ between observations;
6: a noise rate parameter $\epsilon \in [0, 1[$.

Output: $\underset{f \in \mathbb{F}(\mathcal{G})}{\arg\min} \sum_{k=1}^{q} l_k$

such that:

$$\forall\, \mathcal{T}_k \in \{\mathcal{T}_1, \cdots, \mathcal{T}_q\},\ \exists\, \{(v^j, w^j, x^j)\}_{j=1}^{l_k} \in \mathrm{rFBA}(\mathcal{N}, \mathcal{P}, f),\ \forall\, 1 \leq i < |\mathcal{T}_k|,$$
$$|\mathcal{T}_k| \leq l_k \leq |\mathcal{T}_k| + K_{\max} \tag{V.3a}$$

$$\wedge\ 0 < g_k(i+1) - g_k(i) \tag{V.3b}$$

$$\wedge\ (v^{g_k(i)}, w^{g_k(i)}, x^{g_k(i)}) \text{ and } (\hat{v}^i, \hat{w}^i, \hat{x}^i) \text{ are data-compatible} \tag{V.3c}$$

$$\wedge\ \frac{\hat{v}^i_{\mathrm{growth}}}{1+\epsilon} \leq v^{g_k(i)}_{\mathrm{growth}} \tag{V.3d}$$

$$\wedge\ \forall\, v' \in \mathrm{rMSS}(\mathcal{N}, w^{g_k(i)}, x^{g_k(i)}),\ v'_{\mathrm{growth}} \leq \frac{\hat{v}^i_{\mathrm{growth}}}{1-\epsilon} \tag{V.3e}$$

where $g_k : [0, |\mathcal{T}_k|] \to [0, l_k]$ is a bijective function mapping observations of the observed time series $\mathcal{T}_k$ to RMSSs of the trace $\{(v^j, w^j, x^j)\}_{j=1}^{l_k}$.

CEGAR-based solving framework. In this chapter, we introduce a generic hybrid solving framework to solve any OPT+qLP problems. This generic framework is based on the *Counter-Example Guided Abstract Refinement* (CEGAR) framework (Clarke et al., 2003),

The CEGAR framework relies on the same principles as *MERRIN*'s framework (Chapter IV). A Boolean over-approximation of the OPT+qLP problem is solved with a SAT or ASP solver. If this Boolean abstraction is unsatisfiable, then the OPT+qLP problem is unsatisfiable too. Otherwise, a model of the Boolean abstraction is found. This model is a solution to the OPT+qLP problem if it satisfies the quantified linear constraints. Otherwise, it is a counter-example, and the abstraction is refined with additional constraints derived from the counter-example. This iterative process continues until either the OPT+qLP problem is proven to be unsatisfiable or all its models have been enumerated.

The counter-example generalization used in Chapter IV is based on a monotone property on optimal metabolic fluxes for sets of inhibited reactions. In practice, adding an inhibition on a reaction r can be seen as adding a new linear constraint that fixes the metabolic flux of r to 0, *i.e.* adding the constraint $v_r = 0$ to the FBA. Consequently, we extend this property over optimal metabolic fluxes to a monotone property over the optimum values of linear optimization problems for sets of linear constraints (`Property 4 in the paper`).

Outlines. It's crucial to emphasize that the CEGAR-based solving framework presented in this chapter is not just a mere formalization of the solving method introduced in Chapter IV. In addition to introducing a generic framework for efficiently solving OPT+qLP problems, we introduce novel contributions. In the next sections, we present two of these contributions: the generalization of core conflicts (Section 2.1), and a linear quantifier elimination method (Section 2.2). By encoding the OPT+qLP inference problem with *MerrinASP*, our implementation of the CEGAR-based solving framework, we achieve scalability for medium-scale instances on the inference problem (Section 3).

2 Contributions of AAAI's paper

2.1 Core Conflicts Generalization

Overview of the core conflicts and constraints generations described in Section `Counter-Examples Generalization` *of the paper. Explanations are given regarding the inference problem.*

For the rest, let f be a Boolean network (BN) and $\mathcal{R}_f$ be the set of reactions participating in f, *i.e.* the set of reactions that are regulated or that influenced a regulatory rule. Let $\mathcal{R}_i \subseteq \mathcal{R}_f$ be a set of inhibited reactions at time i.

Challenge. In Chapter IV, we introduce a dedicated hybrid solving framework for the flux-based inference problem, and its implementation *MERRIN*. This dedicated framework uses counter-example generalization to prevent spurious candidate BNs from being generated by the ASP solver. In particular, it generates new constraints for each spurious Boolean metabolic steady-state (BMSS), *i.e.* not satisfying the FBA or the phenotype-compatibility constraint, associated with the spurious BNs.

Given i the timestep of a spurious BMSS and its sets of inhibited reactions $\mathcal{R}_i$, the generated constraints prohibit either any subsets of $\mathcal{R}_i$, if the optimum growth is greater than the observation, or any supersets of $\mathcal{R}_i$, if the optimum growth is lesser than the observation. While this method enables *MERRIN* to scale on the model of core-carbon metabolism, it does not filter spurious solutions sufficiently to scale to larger models. Therefore, it is necessary to enhance the constraint generalization method to reduce the number of spurious candidate solutions generated.

> ***Challenge***
> Improving the constraint generalization method to better filter spurious candidate BNs.

Conflicts. There are two types of linear conflict: existential conflict and universal conflict. According to the inference problem, the former corresponds to an observation for which there are no metabolic steady-states that satisfy the observed growth or the FBA equations. It fails to satisfy Eq. V.3d of the OPT+qLP inference problem. Existential conflicts are generalized by prohibiting subsets of inhibited reactions from being selected. The latter is an observation for which the optimal growth allowed by the regulatory state is greater than the observed growth. It fails to satisfy Eq. V.3e. Universal conflicts are generalized by prohibiting supersets of inhibited reactions from being selected.

Unsatisfiable cores. A set of inhibited reactions $\mathcal{R}_i$ has $2^{|\mathcal{R}_f \setminus \mathcal{R}_i|}$ supersets. The smaller the set of inhibited reactions, the more supersets it has. Therefore, when we generalize an existential conflict, we want to generate constraints from the smallest set of inhibited reactions. Given $\mathcal{R}_i$, an *unsatisfiable core* is a smallest subset of $\mathcal{R}_i$ that still induces an existential conflict. Unsatisfiable cores are widely used in modern *Satisfiability Modulo Theory* (SMT) solvers to generalize existential conflicts (Cimatti et al., 2011; Khasidashvili et al., 2015; Zeljić et al., 2017). The constraint generated by an unsatisfiable core is defined in `Eq. 6 of the paper`.

There are no efficient methods to compute minimum unsatisfiable cores. A naive, and widely spread, approach consists of iterating over each inhibited reaction. If removing the reaction still leads to an existential conflict, then the reaction is removed. Otherwise, the reaction is kept. At the end of the process, the remaining reactions form an unsatisfiable core. This method does not guarantee to generate the smallest unsatisfiable core. However, it allows computing an unsatisfiable core by sequentially solving $|\mathcal{R}_i|$ linear optimization problems.

Optimal cores. Based on unsatisfiable cores, we introduce *optimal cores* to generalize universal conflicts. An optimal core of a set of inhibited reactions $\mathcal{R}_i$ is the largest superset of $\mathcal{R}_i$ that still leads to universal conflict. The larger the optimal core, the more it has subsets, and the more spurious candidate it filters. By `Property 4`, if an optimal core leads to a universal conflict, all its subsets will also lead to universal conflicts. The constraint generated by an optimal core is defined in `Eq. 7 of the paper`.

Computing an optimal core from a set of inhibited reactions $\mathcal{R}_i$ is made by sequentially solving $|\mathcal{R}_f| - |\mathcal{R}_i|$ linear optimization problems. The idea is to iterate over each reaction $r \in \mathcal{R}_f \setminus \mathcal{R}_i$ that is not inhibited. If adding the reaction still leads to a universal conflict, then the reaction is added $\mathcal{R}_i$. At the end of the process, $\mathcal{R}_i$ is an optimal core. As for unsatisfiable cores, this method does not guarantee to generate the largest optimal core.

***Solution* – in short**
Given a conflicting set of inhibited reactions $\mathcal{R}_i$, the new constraints are generated from core conflicts: unsatisfiable cores or optimal cores. Unsatisfiable cores are minimal subsets of $\mathcal{R}_i$, and optimal cores are maximal supersets of $\mathcal{R}_i$ according to $\mathcal{R}_f$. Core conflicts can be linearly computed from $\mathcal{R}_i$ by sequentially solving at most $|\mathcal{R}_f|$ linear optimization problems. They increase the number of spurious solutions filtered by the counter-example generalization.

In practice. Interestingly, while computing core conflicts needs to solve numerous linear optimization problems, they still represent huge performance gains. In practice, we notice fewer calls to linear solvers when using core conflicts than without them. Indeed, preventing the generation of spurious solutions is more efficient than solving hundreds of linear optimization problems.

Regarding the inference problem, core conflicts allow for scaling on medium-scale regulated metabolic networks. For instance, no BNs were inferred in 24 hours on instances of the medium-scale regulated metabolic network (RMN) without the use of core conflicts, while it took less than 2 minutes to infer a first BN with them.

2.2 Linear Quantifier Elimination

Description of the linear quantifier elimination used to generate quantifier-free instances of the inference problem in Section `Results` *of the paper.*

Satisfiability Modulo Theory solvers. To figure out if our CEGAR-based solving method is the best solving method for OPT+qLP problems, we need to compare its performances against other hybrid solvers. Recall that the inference problem is a Satisfiability Modulo Theory (SMT) problem that merges optimization, logical constraints, and quantified linear constraints. Although there are SMT solvers, like *z3* (De Moura and Bjørner, 2008), that can theoretically handle optimization, logical constraints, and quantified linear constraints, we found that they do not support problems with both a Boolean objective function to optimize and universal quantifiers. With *z3*, we get warning messages about optimization with quantified constraints not being supported. They are several posts on GitHub[1] and StackOverflow[2] about these issues. Moreover, SMT solvers are not as efficient in solving highly combinatorial problems as ASP-based solvers (Gebser et al., 2014). We confirm these last results on simplified instances of the inference problem, by only keeping the logical constraints (*i.e.* we remove the optimization constraints and quantified linear constraints): ASP was about 10 times faster than *z3* to find a first solution. SMT solvers are therefore not adapted to solve the inference problem efficiently.

To the best of our knowledge, there are no ASP-based solvers that handle quantified linear constraints. However, there exist ASP-based solvers that handle quantifier-free linear constraints, such as *clingo-lpx* (Janhunen et al., 2017). *Clingo-lpx* extends the ASP solver *clingo* (Gebser et al., 2017) with an incremental implementation of the simplex algorithm (Dutertre and De Moura, 2006). To compare our solving framework with *clingo-lpx*, we need to convert the inference problem into an optimization problem under quantifier-free linear constraints (OPT+LP).

> ***Challenge***
> Defining a linear quantifier elimination method to convert the quantified linear constraints into equivalent quantifier-free linear constraints.

Linear optimization problem. As described in `Theorem 2 of the paper`, ensuring the satisfiability of universally quantified constraints comes down to ensuring

[1] https://github.com/Z3Prover/z3/issues/6941
[2] https://stackoverflow.com/questions/59363694/quantifiers-with-maxsmt-in-z3

maximize $c^T \cdot z$	minimize $b^T \cdot y$
such that $A \cdot z \leq b$	such that $A^T \cdot y = c$
with $z \in \mathbb{R}^p$	with $y \in \mathbb{R}^{q+}$
(a) Primal formulation	**(b)** Dual formulation

Figure 16 – Primal (a) and dual (b) formulations of the linear optimization problem (A, b, c). If the primal formulation has p variables and q constraints, its dual formulation has q variables and p constraints.

constraints over optimal values of linear optimization problems. Therefore, transforming a linear optimization problem into an equivalent linear satisfiability problem will allow the elimination of universal linear quantifiers.

For the rest, we model a linear optimization problem over p variables and q constraints as a triplet (A, b, c)[3] with **(i)** A the $q \times p$ matrix of linear constraint coefficients, **(ii)** $b \in \mathbb{R}^q$ the constraints upper bounds, and **(iii)** $c \in \mathbb{R}^p$ the linear coefficient of the objective function. Any linear optimization problem (A, b, c) has a primal formulation and a dual formulation, described by Fig. 16a and Fig. 16b, respectively. While the primal formulation has p variables and q constraints, the dual formulation has q variables and p constraints. The dual formulation is built such that each variable (*resp.* constraint) of the primal formulation becomes a constraint (*resp.* variable) in the dual formulation.

Duality theorems. For linear optimization problems, both the weak and the strong duality theorems hold. The **weak duality theorem** states that the objective value of all feasible solutions of the primal formulation is lesser or equal to the objective value of all feasible solutions of the dual formulation, *i.e.* $\forall z \in \mathbb{R}^p, \forall y \in \mathbb{R}^{q+}, (A \cdot z \leq b \wedge A^T \cdot y = c) \implies c^T \cdot z \leq b^T \cdot y$. In particular, if the primal formulation is unbounded, then the dual formulation will be unsatisfiable. If the primal formulation is not satisfiable, then the dual formulation can either be unbounded or unsatisfiable. Moreover, the **strong duality theorem** states that if the primal formulation has an optimal solution, then the dual formulation has an optimal solution too and that both optimums are equal.

Linear quantifier elimination. If the primal formulation of (A, b, c) has an optimal value, *i.e.* is satisfiable and bounded, its dual formulation has an optimal value too. Let $\Lambda \in \mathbb{R}$ be an upper bound of the optimal value of the primal formulation. From the duality theorem, ensuring that $\forall z \in \mathbb{R}^p, (A \cdot z \leq b) \implies (c^T \cdot z \leq \Lambda)$

[3] This notation for linear optimization problems is equivalent to the one used in the paper.

comes down to checking if a solution of the dual formulation has an objective value lesser or equal to Λ, *i.e.* if $\exists\, y \in \mathbb{R}^{q+}, (A^T \cdot y \leq c) \wedge (b^T \cdot y \leq \Lambda)$. Therefore, the satisfiability of universally quantified linear constraints can be checked by solving linear satisfiability problems rather than linear optimization problems.

Given $x \in \mathbb{B}^n$ Boolean-valued variables, let's consider the universally quantified linear constraint $\phi_\forall$ defined such that (`Eq. 1.c of the paper`):

$$\phi_\forall = \forall\, z \in \mathbb{R}^p, \bigwedge_{e \in E} e(x,z) \implies \bigwedge_{h \in H} h(x,z) \tag{V.4}$$

where E and H denote sets of hybrid clauses of the form "$e(x,z) = \bigvee_i x_{ei} \bigvee_j \neg x_{ej} \vee f_e(z) \leq \Lambda_e$" with f_e denoting linear functions over reals of the form "$\sum_{k=0}^{p} \lambda_{ei} \times z_i$" with $\lambda_{ei} \in \mathbb{R}$. Let A_E be the $|E| \times p$ matrix of linear coefficient $(\forall\, e \in E, \forall\, i \in [0,n], A_{E(ei)} = \lambda_{ei})$, $b_E \in \mathbb{R}^{|E|}$ the linear constraints upper bounds $(\forall\, e \in E, b_{Ee} = \Lambda_e)$, and $c_e \in \mathbb{R}^p$ the linear coefficients of f_e $(\forall\, i \in [0,p], c_{ei} = \lambda_{ei})$.

If $A_E \cdot z \leq b_E$ is satisfiable, then a quantifier-free formula ϕ_{QF} equivalent to $\phi_\forall$[4] can be defined such that (Eq V.5):

$$\phi_{\text{QF}} = \bigwedge_{h \in H} A_E^T \cdot y_h = c_h \wedge y_h \geq 0 \tag{V.5a}$$

$$\wedge \left(\bigvee_i x_{hi} \bigvee_j \neg x_{hj} \vee b^T \cdot y_h \leq \Lambda_h\right) \tag{V.5b}$$

$$\wedge \bigwedge_{e \in E} \left(\neg\left(\bigvee_i x_{ei} \bigvee_j \neg x_{ej}\right) \vee y_{he} = 0\right) \tag{V.5c}$$

where $\forall\, h \in H, y_h \in \mathbb{R}^{|E|}$. The dual constraints are represented by Eq V.5a, and the hybrid clauses $h(x,z)$ by Eq V.5b. The dependencies between the Boolean-valued variables and the universally quantified linear constraints are modeled by Eq. V.5c. Given an hybrid clause $e(x,z) \in E$, if $x \in \mathbb{B}^n$ is a model of $e(x,z)$ $(x \models e(x,z))$ then the linear constraint $f_e(z) \leq \Lambda_e$ does not need to be satisfied. The linear constraint is not added to the primal formulation of the underlying linear optimization problem solved to ensure the universally quantified constraints. Therefore, there is no dual variable y_{he} associated with $f_e(z) \leq \Lambda_e$ in the dual formulation. Removing the dual variable y_{he} from the dual formulation is equivalent to fixing it to 0.

In practice, we manually applied this quantifier elimination method. No automated process has been developed.

Example. Let's consider the OPT+qLP problem ψ used as example in the paper (described in `Fig. 1`). The problem is described in more detail in Section `Combinatorial Optimization Problems Modulo Quantified Linear Constraints`.

[4] If $A_E \cdot z \leq b_E$ is not satisfiable, ϕ_{QF} is an over-approximation of ϕ, *i.e.* $\phi_{\text{QF}} \implies \phi$.

$$\begin{aligned}&\text{maximize} \quad z_2\\&\text{such that:}\\&\quad -z_2 \leq -1\\&\quad z_1 + z_2 \leq 1\\&\quad -z_1 + z_2 \leq 0\\&\text{with } z_1, z_2 \in \mathbb{R}\end{aligned}$$

(a) Primal formulation

$$\begin{aligned}&\text{minimize} \quad -a + b\\&\text{such that:}\\&\quad b - c = 0\\&\quad -a + b + c = 1\\&\text{with } a, b, c \in \mathbb{R}^+\end{aligned}$$

(b) Dual formulation

$$\begin{aligned}\psi_{\text{QF}} = (x_1 \vee x_2 \vee x_3)& \\ \wedge (-a + b \leq 0.6)& \quad \text{(V.6a)}\\ \wedge (b - c = 0) \wedge (-a + b + c = 1)& \quad \text{(V.6b)}\\ \wedge (a = 0 \vee x_1) \wedge (b = 0 \vee x_2) \wedge (c = 0 \vee x_3)& \quad \text{(V.6c)}\\ \text{with } x \in \mathbb{B}^3,\ a, b, c \in \mathbb{R}^+&\end{aligned}$$

(c) Quantifier-free version of ψ

Figure 17 – Application of the linear quantifier elimination method on the example OPT+qLP problem ψ described in `Figure 1` of the paper. (a), (b) are the primal and dual formulations of the associated linear optimization problem, respectively. (c) is the quantifier-free OPT+qLP problem obtained by applying quantifier elimination on ψ.

The underlying linear optimization problem consists in maximizing z_2 under the three linear constraints: $-z_2 \leq -1$, $z_1 + z_2 \leq 1$ and $-z_1 + z_2$ with $z_1, z_2 \in \mathbb{R}$ real-valued variables (Fig. 17a). Its dual formulation is shown in Fig. 17b, it has two linear constraints ($b - c = 0$ and $-a + b + c = 1$) and three variables ($a, b, c \in \mathbb{R}^+$).

Based on Eqs. V.5, we can eliminate the universal linear quantifier and rewrite ψ as ψ_{QF} (Fig 17c). The formula ψ_{QF} contains 6 linear constraints and no universally quantified linear constraints. It has 3 Boolean-valued variables (x_1, x_2, x_3) and 3 real-valued variables a, b, c. Equation V.6a models the universally quantified linear constraints $z_2 \leq 0.6$ of ψ. The two linear constraints of the dual formulation are described in Eq. V.6b. Finally, Eq. V.6c describes 3 hybrid clauses modeling the impact of the Boolean-valued variables on the primal formulation constraints.

There are two assignments of the Boolean-valued variables that satisfy ψ: $\{x_2, x_3\}$ and $\{x_1, x_2, x_3\}$. For both Boolean-valued variable assignments, the assignment $a = 0, b = 0.5, c = 0.5$ is a model of ψ_{QF}. They are the only two Boolean-valued assignments satisfying ψ_{QF}. If x_2 (*resp.* x_3) is not in the assignment, then the linear constraints $b = 0$ (*resp.* $c = 0$), $b - c = 0$, and $-a + b + c = 1$

are not satisfiable. The formulas ψ and ψ_{QF} are equivalent.

> ***Solution*** **– in short**
> We propose a quantifier elimination method based on the strong duality theorem. This quantifier elimination method can be applied to transform any universally quantified linear constraints (Eq. V.4) into a quantifier-free set of linear constraints (Eqs. V.5). If the set of all linear constraints in the left-hand part of universally quantified linear constraint is satisfiable, then both formulas are equivalent. Otherwise, the quantifier-free formula is an over-approximation of the quantified formula.

Application to the OPT+qLP abstraction of the inference problem. The linear quantifier elimination was manually applied to the OPT+qLP inference problem to generate quantifier-free instances for our benchmarks. These quantifier-free instances are equivalent to the quantified ones, as the underlying linear optimization problems are always satisfiable. For all the RMNs considered in this manuscript, the null metabolic flux is always a solution to the FBA equations[5].

3 Complementary Benchmarking and Discussion

This section extends the paper's discussion and discusses the remaining bottlenecks regarding the solving of the inference problem.

3.1 Summary of AAAI's Paper

In this chapter, we introduce a CEGAR-based solving framework for addressing optimization problems under logical and quantified linear constraints (OPT+qLP). The linear constraints are restricted to one level of linear quantifiers. This framework relies on monotone properties over the optimal values of linear optimization problems, and refinements of counter-examples using core conflicts (Section 2.1) to generalize counter-examples. We implement this CEGAR-based framework in *MerrinASP*[6]. It extends the ASP solver *clingo* with linear constraints with one level of quantifier and extends the ASP syntax to model quantified linear constraints. *MerrinASP* is the first ASP-based solver to natively handle quantified linear constraints.

Along with the CEGAR-based solving framework, we introduce a quantifier elimination method (Section 2.2). In the general case, converted quantifier-free problems are not guaranteed to be equivalent to the original OPT+qLP problems.

[5] This assertion does not hold whenever a reaction is forced to have a non-zero metabolic flux.
[6] Available on GitHub: https://github.com/kthuillier/merrinasp

Instance	Search space size	All networks #	All networks Time (s)	Subset minimal networks #	Subset minimal networks Time (s)
Toy	$O(10^6)$	4	1.5	1	1
Core	$O(10^{15})$	48	7	1	7
Medium-scale	$\Omega(10^{380})$	168 861*	86 400*	168 861*	86 400*

* *Number of BNs inferred in 24 hours. Not all solutions are enumerated.*

Table 7 – Performances of the *MerrinASP* implementation of the OPT+qLP inference problem on three regulated metabolic models (described in Chapter II). Times are given as an approximated order of magnitude in seconds. All results are given for complete data (undegraded kinetics, fluxomics, and transcriptomics observations).

However, for the specific OPT+qLP formulation of the inference problem, both problems are equivalent. The quantifier elimination allows for generating quantifier-free instances of the OPT+qLP inference problem that can be solved with *clingo-lpx*, a state-of-the-art ASP modulo quantifier-free linear arithmetic solver.

Benchmark. We evaluate the performance of *MerrinASP* against *clingo-lpx* with quantifier elimination. In practice, we found our implementation, *MerrinASP*, to be 10 times more efficient in solving OPT+qLP problems than *clingo-lpx* with quantifier elimination.

Moreover, to highlight the impact of optimal cores on the solving process and mitigate implementation bias, we compare the performance of *MerrinASP* on both the quantified and quantifier-free instances of the inference problem. On average, solving the quantified instances is 20 times faster than solving the quantifier-free instances. Moreover, on large-scale instances, quantified instances necessitate 7 times fewer calls to linear solvers than quantifier-free instances. These results highlight the advantages of utilizing optimal cores for counter-example generalization in solving OPT+qLP problems, as opposed to relying solely on quantifier elimination. These results are summarized in `Table 2 in the paper`.

3.2 Performance on OPT+qLP Inference Problem Instances

We have encoded the OPT+qLP inference problem with the extended ASP syntax of *MerrinASP*. The MerrinASP encoding is available in Appendix B.3. Table 7 summarizes the number of inferred BNs and the computation times[7] on the three regulated metabolic networks considered in this manuscript. All results discussed in this section are for complete time series observations, *i.e.* noise-free time series with kinetics, fluxomics, and transcriptomics observations.

[7]Fedora 34 with an 8-cores processor i7-1165G7@2.80 GHz and 16GB of RAM

Performance on small-scale instances. For the toy and the core-carbon models, this new implementation reduces the computation time by a factor of 5 when enumerating all BNs compared to the dedicated implementation (described in Chapter IV). There is no difference in the computation time when only enumerating subset-minimal BNs.

Scalability. For the medium-scale model, 168 861 BNs have been inferred in 24 hours while no model was inferred in 24 hours with the dedicated implementation. In particular, it now took about 2 minutes to infer a first BN. The scalability and efficiency of the CEGAR-based framework come from the use of core conflicts. Core conflicts allow for an efficient generalization of counter-examples, and thus an efficient filtering of spurious candidate solutions. In particular, core conflicts are a key element in the scalability of our solving framework. Recall that the dedicated solving framework introduced in Chapter IV does not rely on core conflicts.

3.3 Limits: Enumerating all the solutions

Enumeration of BNs. As described in Table 7, 168 861 BNs were inferred within 24 hours for the medium-scale model. Despite this substantial number of inferred networks, it is important to note that not all possible BNs were enumerated. It is therefore necessary to identify the limiting factors of our CEGAR-based solving framework regarding the solving of the OPT+qLP inference problem.

Redundant operations. Our solving framework is based on the CEGAR framework which is known to be efficient in proving the satisfiability of a problem but not in enumerating solutions (Brummayer and Biere, 2009; Lagniez et al., 2017). The linear checks are processed even for candidate solutions that are sure to satisfy them.

Part of the complexity of the inference problem lies in finding the rFBA traces compatible with the observed time series. In practice, many BNs can be compatible with the same set of rFBA traces. With our CEGAR-based framework, linear checks are made for each rFBA trace even if this trace has already been shown valid for another candidate BN. In particular, all the 168 861 inferred BNs are associated with the same set of rFBA traces.

It should therefore be needed to rework the encoding and solving workflow of the OPT+qLP inference problem. For instance, one could imagine only enumerating BNs associated with distinct rFBA traces and then computing all equivalent BNs (BNs having the same rFBA traces). Since the rFBA traces would have already been validated, no linear checks would be needed for the equivalent BN enumeration, which should reduce computation costs.

CEGAR-based approach for solving combinatorial optimization modulo quantified linear arithmetics problems

Kerian Thuillier[1], **Anne Siegel**[1], **Loïc Paulevé**[2]

[1] Univ. Rennes, Inria, CNRS, IRISA, UMR6074, F-35000 Rennes, France
[2] Univ. Bordeaux, Bordeaux INP, CNRS, LaBRI, UMR5800, F-33400 Talence, France
kerian.thuillier@irisa.fr, anne.siegel@irisa.fr, loic.pauleve@labri.fr

Abstract

Bioinformatics has always been a prolific domain for generating complex satisfiability and optimization problems. For instance, the synthesis of multi-scale models of biological networks has recently been associated with the resolution of optimization problems mixing Boolean logic and universally quantified linear constraints (OPT+qLP), which can be benchmarked on real-world models. In this paper, we introduce a Counter-Example-Guided Abstraction Refinement (CEGAR) to solve such problems efficiently. Our CEGAR exploits monotone properties inherent to linear optimization in order to generalize counter-examples of Boolean relaxations. We implemented our approach by extending *Answer Set Programming* (ASP) solver CLINGO with a quantified linear constraints propagator. Our prototype enables exploiting independence of sub-formulas to further exploit the generalization of counter-examples. We evaluate the impact of refinement and partitioning on two sets of OPT+qLP problems inspired by system biology. Additionally, we conducted a comparison with the state-of-the-art ASP solver *Clingo[lpx]* that handles non-quantified linear constraints, showing the advantage of our CEGAR approach for solving large problems.

Introduction

Satisfiability (SAT) solving has proven to be highly successful in addressing a wide range of real-world combinatorial satisfiability problems across various fields. In the last decades, many applications in bioinformatics have been formulated as complex combinatorial satisfiability and optimization problems according to biological knowledge and data. For decision-aided tasks, life-scientists then take advantage of sampling the full space of solutions in order to prioritize future experiments. Therefore, challenges reside both in solving such complex combinatorial problems on large-scale and real-world instances but also in enumerating part, if not all, the set of solutions.

Traditionally, the problems addressed in life-sciences were either linear programming and optimization (LP) problems (Orth, Thiele, and Palsson 2010; von Kamp and Klamt 2014) or Boolean optimization problems (Videla et al. 2017; Chevalier et al. 2019). In this case, efficient approaches based on *Answer Set Programming* (ASP), a logic programming framework for symbolic satisfiability problems (Baral 2003), have been developed. They take advantage of the ability of modern ASP solvers, like *Clingo* (Gebser et al. 2017), to support various reasoning modes, Boolean optimization, and model enumeration.

A recent evolution in life-sciences is the emergence of hybrid optimization problems combining Boolean logic and linear constraints (Frioux et al. 2019; Mahout, Carlson, and Peres 2020). ASP solvers handling quantifier-free linear constraints, like *Clingo[lpx]* (Janhunen et al. 2017), have been developed to solve such hybrid optimization problems, by extending ASP solver with a DPLL-adapted simplex algorithm (Dutertre and De Moura 2006) used by modern *Satisfiability Modulo Theory* (SMT) solvers. A new class of complexity appeared recently with the problem of inferring metabolic regulatory rules, which is formulated as a hybrid optimization problem with one level of *quantified* linear constraints (Thuillier et al. 2022) and associated with real-world benchmarks. The goal of this paper is to investigate efficient solutions to solve this new class of hybrid optimization problems, which we denote as OPT+qLP.

The state-of-the-art strategy to solve OPT+qLP problems is to rely on quantifier elimination to get back to quantifier-free hybrid optimization problems. There is an equivalence between universally quantified linear constraints and constraints on the optimum of LP problems. Hence, based on the *strong duality theorem*, universally quantified linear constraints can be converted into equi-satisfiable quantifier-free linear constraints through a dual transformation. This allows tackling OPT+qLP problems with standard hybrid approaches, as offered by *Clingo[lpx]* and SMT solvers.

An alternative lies in the *Counter-Example-Guided Abstraction Refinement* (CEGAR) method (Clarke et al. 2003). While sharing similarities with the DPLL algorithm (Nieuwenhuis, Oliveras, and Tinelli 2006) used in modern SMT solvers, the CEGAR approach enables to easily compose solvers for different tasks, including for Boolean optimization and enumeration problems. The strength of the CEGAR approach therefore lies in its generic and solver-independent nature, which allows for taking advantage of the structure of linear problems. It has been widely applied for the solving of quantified Boolean formula (Janota et al. 2016), and SMT problems (Brummayer and Biere 2008; Barrett and Tinelli 2018). However, CEGAR approaches have not been applied so far to OPT+qLP problems.

In this paper, we introduce a CEGAR-based algorithm to solve and enumerate models to OPT+qLP problems. Our approach refines Boolean abstraction of the OPT+qLP problem using monotone properties on LP problems structures and linear constraints partitioning. We rely on the resolution of a formula, a Boolean abstraction, that subsumes the models of the OPT+qLP problem. If this abstraction is unsatisfiable, then so is the OPT+qLP problem. Otherwise, a model of the Boolean abstraction is found. This model is a solution to the OPT+qLP problem if it satisfies the quantified linear constraints. Otherwise, it is a counter-example, and the abstraction is refined with additional constraints derived from the counter-example. This iterative process continues until either the OPT+qLP problem is proven to be unsatisfiable or all its models have been enumerated. To implement it, we developed a prototype based on ASP and evaluated its performance on real-world benchmarks based on biological models. Additionally, we conducted a comparison with *Clingo[lpx]* and compared the performance regarding both quantifier elimination and linear constraints partitioning.

Combinatorial optimization problems modulo quantified linear constraints

We focus on combinatorial optimization problems whose constraints merge propositional logic and quantified linear arithmetics (OPT+qLP). The quantified linear constraints are restricted to one level of quantifier. Solving OPT+qLP problems aims at finding variable assignments, or models, satisfying SAT+qLP constraints while minimizing a given objective function.

Let $x \in \mathbb{B}^n$ denotes Boolean variables and $y \in \mathbb{R}^m$ real-valued variables. We consider SAT+qLP formulas of the following form:

$$\bigwedge_{c \in C} c(x) \tag{1a}$$

$$\wedge \quad \bigwedge_{d \in D} d(x, y) \tag{1b}$$

$$\wedge \quad \forall z \in \mathbb{R}^p, \bigwedge_{e \in E} e(x, z) \implies \bigwedge_{h \in H} h(x, z) \tag{1c}$$

where C denotes Boolean clauses of the form $\bigvee_i x_i \bigvee_j \neg x_j$, and D (resp. E, H) denotes hybrid clauses of the form "$\bigvee_i x_i \bigvee_j \neg x_j \vee f(y) \leq 0$" (resp. $\bigvee_i x_i \bigvee_j \neg x_j \vee f(z) \leq 0$), with f denoting linear functions over reals. Given a hybrid clause $c \in D, E, H$, we will denote by f_c its linear constraint $f_c(y) \leq 0$ (resp. $f_c(z) \leq 0$).

Universally quantified linear constraints are modeled by Eq. 1c. The first part of the implication ($\bigwedge_{e \in E} e(x, z)$) defines the domain $\mathbb{D}(x)$ of the universal real-valued variables z according to x. The domain $\mathbb{D}(x)$ is a subset of $\mathbb{R}^p$, and contains all $z \in \mathbb{R}^p$ such that (x, z) satisfy $\bigwedge_{e \in E} e(x, z)$. Eq. 1c is therefore equivalent to $\forall z \in \mathbb{D}(x), \bigwedge_{h \in H} h(x, z)$.

Let ϕ be a SAT+qLP formula of the form of Eq. 1. A variable assignment $(x, y) \in \mathbb{B}^n \times \mathbb{R}^m$ is a model of ϕ if and only if it satisfies ϕ, *i.e.* $(x, y) \models \phi$. The formula ϕ is unsatisfiable, denoted by $\not\models \phi$, if there is no model ν satisfying ϕ. Otherwise, ϕ is satisfiable.

The SAT+qLP satisfiability problem can be extended into an OPT+qLP optimization problem by considering only the models (x, y) of ϕ that minimize an objective function over Boolean variables $g : \mathbb{B}^n \to \mathbb{R}$:

$$\begin{aligned} \text{minimize} \quad & g(x) && \text{(2a)} \\ \text{such that:} \quad & (x, y) \models \phi && \text{(2b)} \\ \text{with} \quad & x \in \mathbb{B}^n, y \in \mathbb{R}^m \end{aligned}$$

For the rest, let (g, ϕ) be an instance of an OPT+qLP problem. A pair $(x, y) \in \mathbb{B}^n \times \mathbb{R}^m$ is a model of (g, ϕ), denoted by $(x, y) \models (g, \phi)$, if and only if Eqs. 2a and 2b are verified.

Many applications can benefit from a comprehensive characterization of the solution space of satisfiability and optimization problems. Thus, in addition to *searching* for a model of an OPT + qLP problem, we will also consider the *enumeration* up to k different models of it.

Example. Let ψ be the SAT+qLP formula of Fig. 1a over Boolean variables x_1, x_2, x_3. It has no existentially quantified real-valued variables and 2 universally quantified real-valued variables z_1, z_2. Using the notations of Eq. 1, ψ has 1 Boolean ($C = \{(x_1 \vee x_2 \vee x_3)\}$) and 4 hybrid clauses ($D = \emptyset$, $E = \{(z_2 \geq 1 \vee \neg x_1), (z_1 + z_2 \leq 1 \vee \neg x_2), (-z_1 + z_2 \leq 0 \vee \neg x_3)\}$, $H = \{(z_2 \leq 0.6)\}$). Fig. 1b gives a graphical representation of the linear constraints.

For the rest, we will write a model ν as a set such that a Boolean variable x_i belongs to ν if and only if $x_i = \top$. Among the 8 models of ψ, only 2 satisfy it: $\nu_1 = \{x_2, x_3\}$ and $\nu_2 = \{x_1, x_2, x_3\}$. For the former, the set of hybrid clauses E is true if and only if at least $z_1 + z_2 \leq 1$ and $-z_1 + z_2 \leq 0$ hold. As shown in Fig. 1b, all assignments of (z_1, z_2) matching these two constraints satisfy $z_2 \leq 0.6$. For the latter, it does not exist an assignment of (z_1, z_2) that satisfies all hybrid clauses in E.

Let $g : \mathbb{B}^3 \to \mathbb{R}$ be an objective function such that $g(x_1, x_2, x_3) = |x_1| + |x_2| + |x_3|$ with $|x_i| = 1$ if $x_i = \top$, 0 else. Let (g, ψ) be an OPT+qLP problem. Its only model is $\{x_2, x_3\}$ ($g(\{x_2, x_3\}) = 2$ and $g(\{x_1, x_2, x_3\}) = 3$).

Contribution: a CEGAR for solving OPT+qLP

We present a CEGAR-based approach for addressing OPT+qLP problems. Algorithm 1 summarizes the overall procedure. First, we define a Boolean abstraction $(g, \phi_{\text{approx}})$ of the OPT+qLP problem (g, ϕ), such that $(g, \phi) \implies (g, \phi_{\text{approx}})$ (line 2, see details below). Next, we introduce two necessary conditions (lines 3 and 4, see details below) to ensure that there exists a model of (g, ϕ) given a model of $(g, \phi_{\text{approx}})$. If at least one of the two conditions fails, then ϕ_{approx} is refined by generalizing the counter-examples that fail them (line 8, see details below). Finally, we propose a quantified linear constraints partitioning method to increase the efficiency of refinement functions.

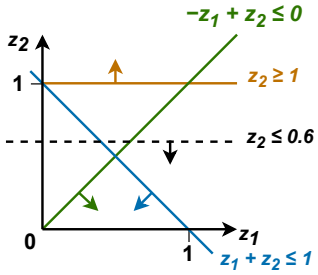
Proofs of the properties, lemmas, and theorems of this section are provided in the technical appendix (Thuillier, Siegel, and Paulevé 2023).

Boolean abstractions of OPT+qLP problems

Let c be a hybrid clause over Boolean variables $x \in \mathbb{B}^n$ and real-valued variables $y \in \mathbb{R}^m$ of the form "$\bigvee_i x_i \bigvee_j \neg x_j \vee$

$$\psi = (x_1 \vee x_2 \vee x_3)$$
$$\wedge\, \forall z \in \mathbb{R}^2, \begin{pmatrix} & (z_2 \geq 1 \vee \neg x_1) \\ \wedge & (z_1 + z_2 \leq 1 \vee \neg x_2) \\ \wedge & (-z_1 + z_2 \leq 0 \vee \neg x_3) \end{pmatrix} \Longrightarrow z_2 \leq 0.6$$

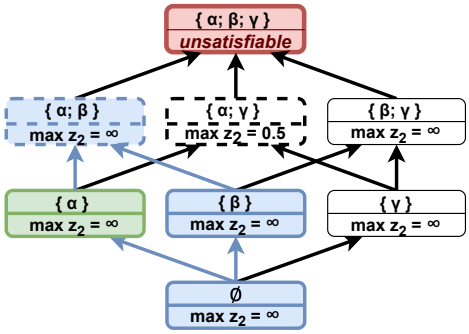
(a) Example SAT+qLP problem ψ.



(b) Visual representation of the quantified linear constraints. No assignments of z_1 and z_2 can satisfy the three linear constraints $z_2 \geq 1$, $z_1 + z_2 \leq 1$ and $-z_1 + z_2 \leq 0$.

$$\psi_{\text{approx}} = (x_1 \vee x_2 \vee x_3)$$
$$\wedge\, (\alpha \vee \neg x_1) \wedge (\beta \vee \neg x_2) \wedge (\gamma \vee \neg x_3) \wedge \delta$$

(c) Boolean abstraction ψ_{approx} of ψ described in **(a)**.



(d) Hasse diagram of all the quantified linear constraints subsets of the example OPT+qLP problem (Fig. 1a) with their optimums. Red block is unsatisfiable. Blocks with dashed borders are the optimal cores of the green block. Blue blocks are the subsets of $\{\alpha; \beta\}$.

Figure 1: Example of SAT+qLP formula ψ **(a)** over three Boolean variables (x_1, x_2, x_3) and two universally quantified real-valued variables (z_1, z_2). Visual representations of the four linear constraints involved in ψ are shown in **(b)**. In **(c)** and **(d)**, $\alpha, \beta, \gamma, \delta$ are Boolean variables associated with the linear constraints $z_2 \geq 1$, $z_1 + z_2 \leq 1$, $-z_1 + z_2 \leq 0$ and $z_2 \leq 0.6$, respectively. The Boolean abstraction ψ_{approx} is defined in **(c)** following Eqs. 4. **(d)** shows the maximum value of z_2 for each subset of linear constraints.

Algorithm 1: CEGAR for solving OPT+qLP problem

Input: an OPT+qLP problem (g, ϕ) of the form Eq. 2
Output: a model $(x, y) \in \mathbb{B}^n \times \mathbb{R}^m$ s.t. $(x, y) \models (g, \phi)$

1: $\phi_{\text{approx}} \leftarrow$ a Boolean abstraction of ϕ of the form Eq. 4
2: **while** $\exists (x, \bar{f}) \models (g, \phi_{\text{approx}})$ **do**
3: **if** $\exists y \models \mathcal{C}_x^D$ **then**
4: **if** $\not\models \mathcal{C}_x^E$ or $\forall h \in \mathcal{C}_x^H, f_h^*(\mathcal{C}_x^E) \leq 0$ **then**
5: **return** x, y
6: **end if**
7: **end if**
8: $\phi_{\text{approx}} \leftarrow \phi_r^{\exists}(x) \wedge \phi_r^{\forall}(x) \wedge \phi_{\text{approx}}$
9: **end while**
10: **return** UNSAT

$f_c(y) \leq 0$". A Boolean abstraction $\bar{c}$ of c is a Boolean clause over the Boolean variables $x \in \mathbb{B}^n$ and $\bar{f}_c \in \mathbb{B}$. The clause $\bar{c}$ is defined by Eq. 3.

$$\bigvee_i x_i \bigvee_j \neg x_j \vee \bar{f}_c \quad \text{denoted by } \bar{c}(x, \bar{f}_c) \tag{3}$$

Let ϕ be a SAT+qLP formula with C its set of Boolean clauses and D, E, H its sets of hybrid clauses. Let $\bar{d}$, $\bar{e}$ and $\bar{h}$ denote Boolean abstractions of the hybrid clauses $d \in D$, $e \in E$ and $h \in H$, respectively. We define the *Boolean abstraction* of ϕ as the following SAT formula:

$$\bigwedge_{c \in C} c(x) \tag{4a}$$
$$\wedge \bigwedge_{d \in D} \bar{d}(x, \bar{f}_d) \tag{4b}$$
$$\wedge \bigwedge_{e \in E} \bar{e}(x, \bar{f}_e) \wedge \bigwedge_{h \in H} \bar{h}(x, \bar{f}_h) \tag{4c}$$

Theorem 1 ($\phi \Rightarrow \phi_{\text{approx}}$)**.** *Let ϕ a SAT+qLP problem and ϕ_{approx} its Boolean abstraction. For any model $(x, y) \in \mathbb{B}^n \times \mathbb{R}^m$ of ϕ, there exists $\bar{f} \in \mathbb{B}^{|D|+|E|+|H|}$ such that $(x, \bar{f})$ is a model of ϕ_{approx}.*

From the above theorem, one can remark that the value $g(x)$ of the objective function on any model (x, y) of an OPT+qLP problem (g, ϕ) is the same on the corresponding model of ϕ_{approx}. In Algorithm 1, the abstraction $(g, \phi_{\text{approx}})$ of the OPT+qLP problem (g, ϕ) is computed line 1. In line 2, the search for $(g, \phi_{\text{approx}})$ models can be performed using a pure Boolean optimization solver. By Theorem 1, if $(g, \phi_{\text{approx}})$ is unsatisfiable, then so is (g, ϕ).

Example. Consider the OPT+qLP problem (g, ψ) from the previous example. Let α, β, γ, δ be four Boolean variables associated with the linear constraints $z_2 \geq 1$, $z_1 + z_2 \leq 1$, $-z_1 + z_2 \leq 0$ and $z_2 \leq 0.6$, respectively. The set of Boolean variables associated with linear constraints is $\bar{f} =$

$\{\alpha, \beta, \gamma, \delta\}$. The Boolean abstraction of ψ is the SAT formula ψ_{approx} defined by Fig. 1c. Formula ψ has two models $\nu_1 = \{x_2, x_3\}$ and $\nu_2 = \{x_1, x_2, x_3\}$. Using the conversion procedure used to prove Theorem 1, $\bar{\nu}_1 = \{x_2, x_3, \beta, \gamma, \delta\}$ and $\bar{\nu}_2 = \{x_1, x_2, x_3, \alpha, \beta, \gamma, \delta\}$ are two models of ψ_{approx}. The model ν_1 is the only model of (g, ψ). It has the optimal score $g^* = 2$. The model $\bar{\nu}_1$ associated with ν_1 has the same score.

Ensuring quantified linear constraints

Let $\mathcal{C}$ be a set of linear constraints of the form $f(y) \leq 0$. A variable assignment $y \in \mathbb{R}^m$ is a model of $\mathcal{C}$, denoted by $y \models \mathcal{C}$, if and only if $y \models \bigwedge_{f \in \mathcal{C}} f(y) \leq 0$. Given $f : \mathbb{R}^m \to \mathbb{R}$ a linear function, $y \in \mathbb{R}^m$ is a model of the *linear optimization problem* $(f, \mathcal{C})$ if and only if $y \models \mathcal{C}$ and it maximizes the objective function f, *i.e.* $\forall y' \in \mathbb{R}^m, y' \models \mathcal{C} \implies f(y') \leq f(y)$. The optimum value of $(f, \mathcal{C})$ will be denoted by $f^*(\mathcal{C}) = \max_{y \models \mathcal{C}} f(y)$.

Let C_h be a set of hybrid clauses and $x \in \mathbb{B}^n$ a Boolean variable assignment. For x to be a model of C_h, it must exist $y \in \mathbb{R}^m$ such that each hybrid clause $h \in C_h$ is satisfied by either x or y. Let $\mathcal{C}_x^{C_h}$ be the set of linear constraints of clauses for which x is not a model:

$$\mathcal{C}_x^{C_h} = \{f_c(y) \leq 0 | c \in C_h, x \not\models c\} \tag{5}$$

Hence, given $c \in C_h$ and $(x, \bar{f}_c) \models \bar{c}(x, \bar{f}_c)$, if $f_c \in \mathcal{C}_x^{C_h}$ then $\bar{f}_c = \top$. Otherwise, x would be a model of $\bar{c}(x, \bar{f}_c)$.

Theorem 2. *Let ϕ be a* SAT+*q*LP *formula and ϕ_{approx} its Boolean abstraction. Given $x \in \mathbb{B}^n$ and $y \in \mathbb{R}^m$, $(x, y) \models \phi$ if and only if the following three conditions hold:* ***(C1)*** *$\exists \bar{f}, (x, \bar{f}) \models \phi_{approx}$;* ***(C2)*** *$y \models \mathcal{C}_x^D$;* ***(C3)*** *$(\not\models \mathcal{C}_x^E) \vee (\bigwedge_{h \in \mathcal{C}_x^H} f_h^*(\mathcal{C}_x^E) \leq 0)$.*

Theorem 2 can be further extended for OPT+qLP problems. Let (g, ϕ) be an OPT+qLP problem and $(g, \phi_{\text{approx}})$ its Boolean abstraction. Any variable assignment $(x, y) \in \mathbb{B}^n \times \mathbb{R}^m$ minimizing g and satisfying *C1*, *C2* and *C3* is a model of (g, ϕ).

Corollary 2.1. *Given $x \in \mathbb{B}^n$ and $y \in \mathbb{R}^m$ a real-valued variables assignment, if* ***(C1')*** *$\exists \bar{f}, (x, \bar{f}) \models (g, \phi_{approx})$,* ***C2*** *and* ***C3*** *hold, then $(x, y) \models (g, \phi)$.*

In Algorithm 1, the condition *C1'* is ensured if a model $(x, \bar{f})$ of $(g, \phi_{\text{approx}})$ is found (line 2). Condition *C2* is ensured in line 3 by finding a model y of the set of linear constraints $\mathcal{C}_x^D$ using a linear programming (LP) solver. *C2* holds only if y exists. Finally, condition *C3* is ensured in line 4. If $\mathcal{C}_x^E$ is satisfiable, a linear optimization problem $(f_h, \mathcal{C}_x^E)$ is solved for each $f_h \in \mathcal{C}_x^H$. The linear optimization problems are solved using LP solvers. Each optimum $f_h^*(\mathcal{C}_x^E)$ is then compared to 0. If at least one optimum is strictly greater than 0, then *C3* does not hold. If the three conditions *C1'*, *C2* and *C3* hold, $(x, y) \models (g, \phi)$ is returned. Otherwise, $(x, \bar{f})$ is a counter-example.

Example. Consider the OPT+qLP problem (g, ψ) and its Boolean abstraction ψ_{approx} (Fig. 1c) from the previous example. The variable assignment $\{x_1, \alpha, \delta\}$ is a model of ψ_{approx} that minimize g, with $g(\{x_1, \alpha\}) = 1$. By Corollary 2.1, $\{x_1\}$ is also a model of (g, ψ) if either $\not\models \{z_2 \geq 1\}$ or if the linear optimization problem $(f_\delta(z_1, z_2) = z_2, \{z_2 \geq 1\})$ has an optimum less or equals to 0.6. From Fig. 1b, we can see that $\{z_2 \geq 1\}$ is satisfiable and that $f_\delta^*(\{z_2 \geq 1\})$ is $+\infty$. Therefore, *C3* does not hold and $\{x_1, \delta\}$ is not a model of (g, ψ). The variable assignment $\{x_1, \alpha, \delta\}$ is a counter-example.

Counter-examples generalization

Let ϕ be a SAT+qLP formula and ϕ_{approx} its Boolean abstraction. Theorem 2 states that for any model $\bar{\nu} = (x, \bar{f})$ of ϕ_{approx} there is a corresponding model ν of ϕ if conditions *C2* and *C3* hold. If either *C2* or *C3* is not satisfied, then $\bar{\nu}$ is a counter-example. From $\bar{\nu}$, new Boolean logic constraints $\phi_r(\bar{\nu})$ can be deduced and used to refine ϕ_{approx}. The new Boolean abstraction of ϕ becomes $\phi_{\text{approx}} \wedge \phi_r(\bar{\nu})$, such that $\phi \implies \phi_{\text{approx}} \wedge \phi_r(\bar{\nu})$.

Existential counter-example. Suppose that $(x, \bar{f})$ does not satisfy *C2*. The set of linear constraints $\mathcal{C}_x^D$ is unsatisfiable, *i.e.* $\not\models \mathcal{C}_x^D$. Therefore, any supersets of linear constraints of $\mathcal{C}_x^D$ will be unsatisfiable too. An *unsatisfiable core* ($\mathcal{C}_{\text{unsat}}$) of a given set of linear constraints $\mathcal{C}$ is the smallest subset of $\mathcal{C}$ for which $\not\models \mathcal{C}_{\text{unsat}}$. In other words, for all $\mathcal{C}' \subset \mathcal{C}_{\text{unsat}}$, there exists a vector $y \in \mathbb{R}^m$ that satisfies $\mathcal{C}'$. When $\mathcal{C}$ is satisfiable, $\mathcal{C}_{\text{unsat}}$ is an empty set. Unsatisfiable cores have been widely used in SMT solvers and CEGAR-based approaches for generalizing sets of unsatisfiable constraints (Cimatti, Griggio, and Sebastiani 2011; Khasidashvili, Korovin, and Tsarkov 2015).

Let $\mathcal{C}_{\text{unsat}}$ be an unsatisfiable core of $\mathcal{C}_x^D$. The refinement function $\phi_r^\exists(x)$ is defined by Eq. 6.

$$\phi_r^\exists(x) = \bigvee_{f \in \mathcal{C}_{\text{unsat}}} \neg \bar{f} \tag{6}$$

Note that refinement function $\phi_r^\exists(x)$ does not generate any constraints if *C2* holds ($\mathcal{C}_{\text{unsat}} = \emptyset$).

Lemma 3. *$\phi \implies \phi_{approx} \wedge \phi_r^\exists(x)$.*

Universal counter-example. Suppose that $(x, \bar{f})$ does not satisfy *C3*. This implies that there is at least one hybrid clause $h \in H$ such that $\mathcal{C}_x^E$ is satisfiable and $f_h^*(\mathcal{C}_x^E) > 0$. Then, any model (x', y') such that $\mathcal{C}_{x'}^E \subseteq \mathcal{C}_x^E$ will be such $f_h^*(\mathcal{C}_{x'}^E) > 0$, as stated by the following property:

Property 4. *Given a linear objective function $\boldsymbol{f}$ and two linear optimization problems $(\boldsymbol{f}, \mathcal{C}_1)$ and $(\boldsymbol{f}, \mathcal{C}_2)$, $\mathcal{C}_1 \subseteq \mathcal{C}_2 \implies \boldsymbol{f}^*(\mathcal{C}_1) \geq \boldsymbol{f}^*(\mathcal{C}_2)$.*

Similarly to unsatisfiable cores, we can introduce the notion of optimal cores. Given a linear objective function f and a set of linear constraints $\mathcal{C}$, an *optimal core* is a biggest superset $\mathcal{C}_{\text{opt}}^f$ of $\mathcal{C}$ such that $\mathcal{C}_{\text{opt}}^f$ is satisfiable and $f^*(C) = f^*(C_{\text{opt}}^f)$.

Let $\mathcal{C}_{\text{opt}}^f$ be an optimal core of $(f, \mathcal{C}_x^E)$. The refinement function $\phi_r^\forall(x)$ is defined by Eq. 7.

$$\phi_r^\forall(x) = \bigwedge_{\substack{h \in \mathcal{C}_x^H \\ f_h^*(\mathcal{C}_x^E) > 0}} \neg \bar{f}_h \vee \bigvee_{\substack{e \in E \\ f_e \notin \mathcal{C}_{\text{opt}}^{f_h}}} \bar{f}_e \tag{7}$$

Lemma 5. $\phi \implies \phi_{approx} \wedge \phi_r^{\forall}(x)$

Constraints generated by the refinement functions $\phi_r^{\exists}(x)$ and $\phi_r^{\forall}(x)$ do not involve the same sets of variables. Therefore, $\phi_r^{\exists}(x) \wedge \phi_r^{\forall}(x) \wedge \phi_{\text{approx}}$ still subsumes ϕ.

Theorem 6. *Given* $(x, \bar{f}) \models \phi_{approx}$, $\phi \implies \phi_r^{\exists}(x) \wedge \phi_r^{\forall}(x) \wedge \phi_{approx}$.

Corollary 6.1. $(g, \phi) \implies \phi_r^{\exists}(x) \wedge \phi_r^{\forall}(x) \wedge \phi_{approx}$.

Corollary 6.2. $\forall \nu^* \models (g, \phi) \implies \exists \nu' \models \phi_r^{\exists}(x) \wedge \phi_r^{\forall}(x) \wedge \phi_{approx}, g(\nu') = g(\nu^*)$.

Algorithm 1 refines the Boolean abstraction ϕ_{approx} in line 8. Corollaries 6.1 and 6.2 ensure that the refined Boolean abstraction is still an overapproximation of (g, ϕ). Therefore, Corollary 2.1 still holds for the next iteration.

Example. Consider ψ_{approx} as defined in Fig. 1c and the counter-example $\{x_1, \alpha, \delta\}$ find previously. This counter-example satisfies *C2* since there are no existentially quantified linear constraints in ψ. Hence, $\phi_r^{\exists}(\{x_1\})$ does not generate any constraints. However, it fails to satisfy *C3*. A Hasse diagram of all the subsets of the set of linear constraints of ψ is shown in Fig. 1d. It can be seen that $\{\alpha\}$ has two optimal cores: $\{\alpha, \beta\}$ and $\{\alpha, \gamma\}$. The set $\{\alpha, \beta, \gamma\}$ is not an optimal core since it is not satisfiable. All linear optimization problems whose linear constraints are either a subset of $\{\alpha, \beta\}$ or of $\{\alpha, \gamma\}$ will also fail *C3*. Suppose that the optimal core $\{\alpha, \beta\}$ has been selected by the refinement function $\phi_r^{\forall}(\{x_1\})$. It will generate the constraints $\neg\delta \vee \gamma$, and it will prohibit selecting any model containing a subset of $\{\alpha, \beta\}$, blue and green boxes in Fig. 1d.

Partitioning quantified linear constraints

Let (g, ϕ) be an OPT+qLP problems with $(g, \phi_{\text{approx}})$ its Boolean abstraction. Linear constraints of ϕ can be partitioned to exploit the sparsity of the underlying linear optimization problems. Let $\mathcal{P} = \{\mathcal{P}_1, ..., \mathcal{P}_k\}$ be a partition of the linear constraints of ϕ such that **(i)** no two linear constraints share variables among different subsets; **(ii)** each subset contains either existentially quantified linear constraints or universally quantified linear constraints.

Let $(x, \bar{f}) \models (g, \phi_{\text{approx}})$. The set of linear constraints $\mathcal{C}_x^D$ can be partitioned in $\mathcal{P}_x^D$ according to the partition $\mathcal{P}$. Deciding the satisfiability of $\mathcal{C}_x^D$ comes down to deciding the satisfiability of each subset $\mathcal{P}_i \in \mathcal{P}_x^D$. If at least one subset is unsatisfiable, so is $\mathcal{C}_x^D$. Otherwise, it exists a model y_i for each subset $\mathcal{P}_i \in \mathcal{P}_x^D$ and $\{y_i\}_i \models \mathcal{C}_x^D$.

Lemma 7. $\exists y \in \mathbb{R}^m, \mathcal{C}_x^D \iff \bigwedge_{\mathcal{P}_i \in \mathcal{P}_x^D} y \models \mathcal{P}_i$.

If $(x, \bar{f})$ fails *C2*, one can exhibit a subset of sets of $\mathcal{P}_x^D$ that are unsatisfiable. Unsatisfiable cores can be computed independently for each unsatisfiable set, which reduces the computational cost of finding unsatisfiable cores. Let $\mathbb{C}_{\text{unsat}}$ be the set of unsatisfiable cores associated with the unsatisfiable sets. The existential refinement function $\phi_r^{\exists}(x)$ can be reformulated as:

$$\phi_r^{\exists}(x) = \bigwedge_{\mathcal{C}_{\text{unsat}} \in \mathbb{C}_{\text{unsat}}} \bigvee_{f \in \mathcal{C}_{\text{unsat}}} \neg \bar{f} \tag{8}$$

Benchmark	*Small-scale*	*Large-scale*
Instances SAT	29	32
Instances UNSAT	31	28
Boolean variables	6.5×10^4	4×10^9
Existential real variables	2×10^3	8×10^3
Universal real variables	2×10^3	8×10^3
Boolean constraints	2.7×10^5	1.8×10^6
Existential linear constraints	6×10^3	25×10^3
Universal linear constraints	6×10^3	25×10^3

Table 1: Benchmarks descriptions. Only the order of magnitude of the number of constraints and variables is given.

Similarly, all linear constraints $f_h \in \mathcal{C}_x^H$ are partitioned with the linear constraints of $\mathcal{C}_x^E$ that can impact their values. Let $\mathcal{P}' \in \mathcal{P}$ be the partitioned containing f_h and $\mathcal{P}_x'^E$ the set of all linear constraints of $\mathcal{C}_x^E$ in $\mathcal{P}'$.

Lemma 8. *If* $\mathcal{C}_x^E$ *is satisfiable, then* $f_h^*(\mathcal{C}_x^E) = f_h^*(\mathcal{P}_x'^E)$.

If $(x, \bar{f})$ fails *C3*, it is necessarily since there is not enough constraints in $\mathcal{P}_x'^E$. Since only linear constraints in $\mathcal{P}'$ have an impact on f_h^*, the computation of an optimal core $\mathcal{P}'_{\text{opt}}$ can be restricted to the set of linear constraints in $\mathcal{P}'$. The universal refinement function $\phi_r^{\forall}(x)$ can be reformulated as:

$$\phi_r^{\forall}(x) = \bigwedge_{\substack{h \in \mathcal{C}_x^H \\ f_h^*(\mathcal{P}_x'^E) > 0}} \neg \bar{f}_h \vee \bigvee_{\substack{e \in E \\ f_e \notin \mathcal{P}'_{\text{opt}}}} \bar{f}_e \tag{9}$$

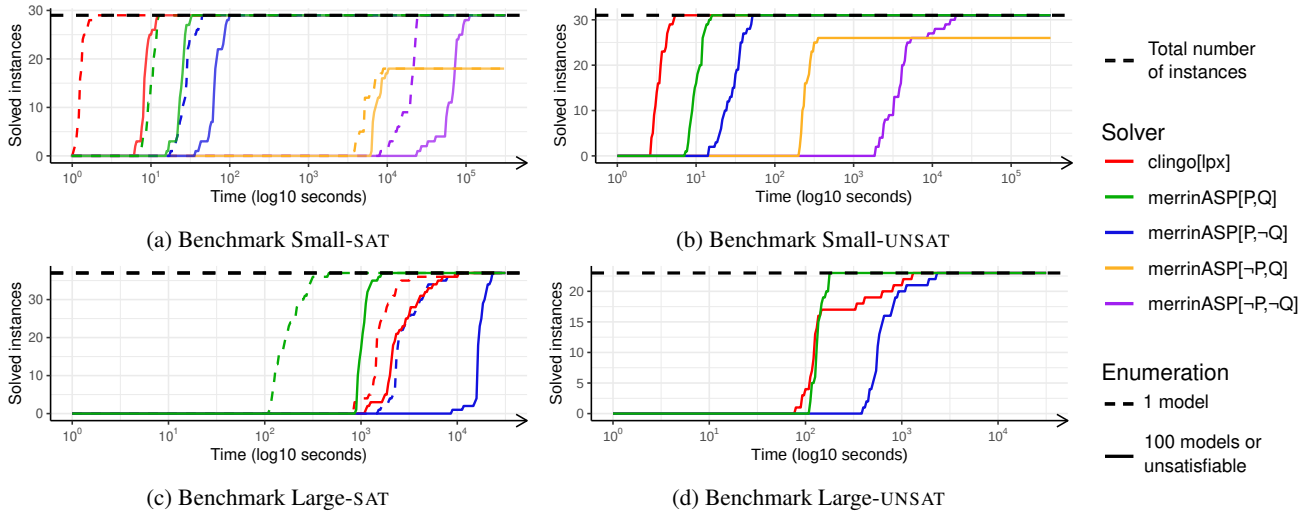
It is important to note that Theorem 6 still holds with these new definitions of $\phi_r^{\exists}$ and $\phi_r^{\forall}$. They generate smaller refinement constraints and allow reducing the computational cost of finding unsatisfiable and optimal cores.

Experiments

We propose MERRINASP (https://github.com/kthuillier/merrinasp), an ASP-based implementation of Algorithm 1. It extends the *Clingo* solver, using its *Python* API, with a linear constraint propagator, implemented with the *Python* PULP library and the LP solver COIN (Lougee-Heimer 2003). Model enumeration is made through the *Clingo* solver which keeps track of all refinements during the enumeration process. The partitioning is explicitly specified in the input problem.

Benchmark

Problem description. Regulatory flux balance analysis (rFBA) is a common model of dynamics of bacteria (Covert, Schilling, and Palsson 2001). The rFBA framework consists in sequentially solving maximum flow problems on weighted hypergraphs. The hyperedge capacities are updated at each step according to Boolean rules. Capacities are either set to 0 or to their initial value. The metabolic regulatory rules inference problem (Thuillier et al. 2022) is an inverse problem. Given a weighted hypergraph and sequences of observed maximum flows, it consists in inferring a set of Boolean rules controlling the hyperedge capacities matching the sequences of observations. For each



(a) Benchmark Small-SAT

(b) Benchmark Small-UNSAT

(c) Benchmark Large-SAT

(d) Benchmark Large-UNSAT

Figure 2: Runtime distribution of 4 configurations of our MERRINASP implementation of the CEGAR-based Algorithm 1 and *Clingo[lpx]* on OPT+QLP problem instances. All variants were applied to a benchmark built from a small-scale real biological model (Figs. (a) and (b), 60 instances) and a large-scale real biological model (Figs. (c) and (d), 60 instances). Small-scale and large-scale benchmarks contain both satisfiable instances (panels (a) and (c)) and unsatisfiable instances (panels (b) and (d)). The four configurations of MERRINASP include a partitioning option (P) and the use of universally quantified linear constraints (Q). Time is given in seconds in $\log_{10}$ scale. Dashed black horizontal lines represent the total number of instances

Benchmark	Partitioned (P)	Quantified (Q)	Deciding SAT Time (s)	Enumeration Time (s)	LP solver Time (s)	Number of LP solvers calls	Number of refinements
Small-SAT	×	×	$18\,761 \pm 4\,759$	$49\,952 \pm 18\,515$	$3\,812 \pm 2\,727$	$16\,795 \pm 2\,364$	2 ± 0
	×	✓	$5\,528 \pm 1\,498$	$2\,116 \pm 1\,044$	$1\,433 \pm 223$	$9\,944 \pm 1\,470$	$\mathbf{1 \pm 0}$
	✓	×	28 ± 6	40 ± 11	34 ± 7	937 ± 111	5 ± 1
	✓	✓	$\mathbf{9 \pm 1}$	$\mathbf{15 \pm 3}$	$\mathbf{15 \pm 2}$	$\mathbf{501 \pm 41}$	6 ± 1
Small-UNSAT	×	×	$5\,143 \pm 4\,395$	NA	$1\,112 \pm 766$	$6\,596 \pm 3\,723$	$\mathbf{1 \pm 0}$
	×	✓	247 ± 38	NA	137 ± 17	$2\,039 \pm 115$	$\mathbf{1 \pm 0}$
	✓	×	30 ± 10	NA	24 ± 10	669 ± 221	9 ± 4
	✓	✓	$\mathbf{10 \pm 2}$	NA	$\mathbf{7 \pm 1}$	$\mathbf{252 \pm 54}$	9 ± 4
Large-SAT	✓	×	$3\,163 \pm 1\,538$	$13\,922 \pm 1\,946$	801 ± 236	$17\,957 \pm 5\,032$	41 ± 16
	✓	✓	$\mathbf{183 \pm 75}$	$\mathbf{865 \pm 112}$	$\mathbf{121 \pm 74}$	$\mathbf{3\,548 \pm 2\,184}$	$\mathbf{21 \pm 11}$
Large-UNSAT	✓	×	739 ± 454	NA	374 ± 248	$7\,480 \pm 4\,673$	17 ± 8
	✓	✓	$\mathbf{135 \pm 19}$	NA	$\mathbf{41 \pm 11}$	$\mathbf{1\,155 \pm 307}$	$\mathbf{13 \pm 3}$

Table 2: Comparative analysis of MERRINASP performance under different configurations. Results are presented as average value $\pm$ standard deviation. Deciding SAT times denote the time needed to find a first model or to decide unsatisfiable. NA indicates information not available. Bold values indicate the best value among all configurations for the current benchmark.

observation, it must find which capacities were set to 0 for the maximum flow to match the observation. In this problem, Boolean clauses delimit admissible Boolean rules according to biological knowledge. For each observation, existential constraints ensure the existence of a corresponding flow, while universal constraints ensure that no flow is strictly higher than the observed one. We refer the reader to the above-mentioned paper for a formal definition of the problem.

Benchmark description. We conducted experiments using MERRINASP on real-world benchmarks of metabolic regulatory rules inference problems (Thuillier, Siegel, and Paulevé 2023). Our benchmarks are composed of 120 instances divided into 60 small-scale instances and 60 large-scale instances. The small-scale benchmark is directly sourced from (Thuillier et al. 2022), while the large-scale benchmark is generated based on a large-scale regulated metabolic network (Covert and Palsson 2002), following the methodology outlined in the aforementioned paper. Benchmarks are described in table 1. Instances of the large-scale benchmarks have approximately 10 times more variables and constraints than instances of the small-scale benchmarks. Linear constraints can be partitioned into about 200

sets for small-scale instances and 140 sets for large-scale instances.

Configuration. Each instance was executed on Haswell Intel Xeon E5-2680 v3 CPU at 2.5GHz and 128GB of RAM and 100 models were enumerated.

Results

We compared MERRINASP with *Clingo[lpx]*, a state-of-the-art ASP solver that handles quantifier-free linear constraints (Janhunen et al. 2017) by extending *Clingo* with a DPLL-adapted simplex algorithm (Dutertre and De Moura 2006). *Clingo[lpx]* supports neither linear constraints partitioning nor universal linear constraints. We further conducted a comparative analysis of MERRINASP under four configurations: with and without partioning of linear constraints (denoted by P and $\neg P$), using the CEGAR approach over quantified linear constraints (denoted by Q) or using quantifier elimination ($\neg Q$). Note that *Clingo[lpx]* is equivalent to the configuration $[\neg P, \neg Q]$, and that MERRINASP$[P, Q]$ exploits all the properties described in previous sections.

Comparison with *Clingo[lpx]*. As shown in Fig. 2a and 2b, on small-scale instances, MERRINASP and *Clingo[lpx]* solve the instances in a similar order of magnitude (10s in average for *Clingo[lpx]* and 30s in average for MERRINASP). On large-scale instances, MERRINASP outperforms *Clingo[lpx]* by a factor of 10 (see Figs. 2c and 2d).

As shown in Fig. 2c, MERRINASP excels at finding the first model in large-scale satisfiable instances, outperforming *Clingo[lpx]* by a factor of 30. The difference in performance between the two solvers heavily depends on the enumeration phase. The CEGAR method requires many checks to ensure that a model of the Boolean abstraction is a model of the original OPT+qLP problem, even after reaching equisatisfiability. Consequently, while MERRINASP is significantly faster than *Clingo[lpx]* in finding the first model for satisfiable problems, both solvers exhibit similar performance in enumerating the other 99 models.

Impact of partitioning (P). Figs. 2a and 2b suggest that linear constraints partitioning (P) increase the performance of MERRINASP by a factor of 1000 on satisfiable instances and a factor of 20 on unsatisfiable instances. No instance of the large-scale benchmark has finished in 48 hours for the not-partitioned configurations. Table 2 shows that while partitioning entails solving a larger number of linear optimization problems, the total number of linear optimization problems solved is reduced by a factor of 10 compared to without partitioning. On the small-scale satisfiable (*resp.* unsatisfiable) instances, MERRINASP$[P, Q]$ solved in average 501 (*resp.* 252) linear optimization problems, against 9 944 (*resp.* 2 039) for MERRINASP$[\neg P, Q]$.

Impact of quantified linear constraints (Q). Our counter-example generation for universally quantified linear constraints consistently outperforms quantifier elimination reformulations by a factor of 3 on the small-scale and 20 large-scale benchmarks. From Table 2, we can see that twice fewer refinements are made when using quantified linear constraints (Q) compared to using quantifier elimination ($\neg Q$). For large-scale (*resp.* small-scale) instances, these refinements were generated using 7 (*resp.* 2) times fewer calls to the linear solvers when using (Q) compared to ($\neg Q$).

Discussion. These results highlight that both linear constraint partitioning (P) and counter-example generation for universally quantified linear constraints (Q) have significant impacts on performance. Using both of them allows dividing computation time by 2 000 compared to not using any of them. They allow for generating more efficient refinements (gain of 2) while reducing the number of linear solver calls (gain of 7). This reduction is attributed to the partitioning approach, which enables solving independent linear optimization problems with a reduced number of constraints and variables. Their small size leads to faster computation of unsatisfiable and optimal cores for each counter-example, and their independence allows for reducing the number of verifications: a set that has passed the linear checks does not have to be checked again.

MERRINASP is a prototype and does not use efficient approaches to instantiate and solve linear optimization problems. In contrast, *Clingo[lpx]* and SMT solvers, such as *z3* (De Moura and Bjørner 2008), use an incremental implementation of the simplex algorithm to check linear constraints (Dutertre and De Moura 2006). Our approach is not dependent on the method used to solve linear constraints. This suggests that MERRINASP has the potential to further enhance its performance by integrating these algorithms.

Conclusion and Future Work

In this paper, we presented a novel approach for solving combinatorial optimization problems with Boolean logic and quantified linear constraints (OPT+qLP), based on Counter-Example-Guided Abstraction Refinement (CEGAR). Our implementation, MERRINASP, was developed using Answer Set Programming.

To evaluate the effectiveness of our approach, we introduced a new benchmark of small-scale and large-scale OPT+qLP problems inspired by systems biology. We compared MERRINASP against a state-of-the-art ASP modulo quantifier-free linear constraints solver, *Clingo[lpx]*. The results highlight that MERRINASP scales significantly better than *Clingo[lpx]* on large-scale satisfiable instances, especially for the search of one model on satisfiable instances. The enumeration of models and unsatisfiable instances remain competitive with *Clingo[lpx]* but suggest room of improvement to improve the CEGAR approach and reduce the number of counter-example checks (Brummayer and Biere 2009; Lagniez et al. 2017).

Looking ahead, we plan to automate the linear constraint partitioning process and explore the integration of our approach with the DPLL-based simplex algorithm used in *Clingo[lpx]*. Moreover, the integration of quantified Linear Real Arithmetics theory (LRA) (Reynolds, King, and Kuncak 2017) could provide complementary refinements using linear constraints, while our approach refines by the means of combinatorial constraints. These future advancements hold the promise of further enhancing the efficiency and applicability of CEGAR-based OPT+qLP solvers.

Acknowledgments

Work of KT and LP is supported by the French Agence Nationale pour la Recherche (ANR) in the scope of the project "BNeDiction" (grant number ANR-20-CE45-0001).

References

Baral, C. 2003. *Knowledge Representation, Reasoning and Declarative Problem Solving*. New York, NY, USA: Cambridge University Press. ISBN 0521818028.

Barrett, C.; and Tinelli, C. 2018. *Satisfiability modulo theories*. Springer.

Brummayer, R.; and Biere, A. 2008. Lemmas on Demand for the Extensional Theory of Arrays. In *Proceedings of the Joint Workshops of the 6th International Workshop on Satisfiability Modulo Theories and 1st International Workshop on Bit-Precise Reasoning*, SMT '08/BPR '08, 6–11. New York, NY, USA: Association for Computing Machinery. ISBN 9781605584409.

Brummayer, R.; and Biere, A. 2009. Effective bit-width and under-approximation. In *International Conference on Computer Aided Systems Theory*, 304–311. Springer.

Chevalier, S.; Froidevaux, C.; Paulevé, L.; and Zinovyev, A. 2019. Synthesis of Boolean Networks from Biological Dynamical Constraints using Answer-Set Programming. In *2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)*, 34–41.

Cimatti, A.; Griggio, A.; and Sebastiani, R. 2011. Computing small unsatisfiable cores in satisfiability modulo theories. *Journal of Artificial Intelligence Research*, 40: 701–728.

Clarke, E.; Grumberg, O.; Jha, S.; Lu, Y.; and Veith, H. 2003. Counterexample-guided abstraction refinement for symbolic model checking. *Journal of the ACM (JACM)*, 50(5): 752–794.

Covert, M. W.; and Palsson, B. Ø. 2002. Transcriptional Regulation in Constraints-based Metabolic Models of Escherichia coli* 210. *Journal of Biological Chemistry*, 277(31): 28058–28064.

Covert, M. W.; Schilling, C. H.; and Palsson, B. 2001. Regulation of gene expression in flux balance models of metabolism. *Journal of theoretical biology*, 213(1): 73–88.

De Moura, L.; and Bjørner, N. 2008. Z3: An efficient SMT solver. In *International conference on Tools and Algorithms for the Construction and Analysis of Systems*, 337–340. Springer.

Dutertre, B.; and De Moura, L. 2006. Integrating simplex with DPLL (T). *Computer Science Laboratory, SRI International, Tech. Rep. SRI-CSL-06-01*.

Frioux, C.; Schaub, T.; Schellhorn, S.; Siegel, A.; and Wanko, P. 2019. Hybrid metabolic network completion. *Theory and Practice of Logic Programming*, 19(1): 83–108.

Gebser, M.; Kaminski, R.; Kaufmann, B.; and Schaub, T. 2017. Multi-shot ASP solving with clingo. *CoRR*, abs/1705.09811.

Janhunen, T.; Kaminski, R.; Ostrowski, M.; Schellhorn, S.; Wanko, P.; and Schaub, T. 2017. Clingo goes linear constraints over reals and integers. *Theory and Practice of Logic Programming*, 17(5-6): 872–888.

Janota, M.; Klieber, W.; Marques-Silva, J.; and Clarke, E. 2016. Solving QBF with counterexample guided refinement. *Artificial Intelligence*, 234: 1–25.

Khasidashvili, Z.; Korovin, K.; and Tsarkov, D. 2015. EPR-based k-induction with Counterexample Guided Abstraction Refinement. In *GCAI*, 137–150.

Lagniez, J.-M.; Berre, D. L.; de Lima, T.; and Montmirail, V. 2017. A Recursive Shortcut for CEGAR: Application To The Modal Logic K Satisfiability Problem. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, 674–680.

Lougee-Heimer, R. 2003. The Common Optimization INterface for Operations Research: Promoting open-source software in the operations research community. *IBM Journal of Research and Development*, 47(1): 57–66.

Mahout, M.; Carlson, R. P.; and Peres, S. 2020. Answer Set Programming for Computing Constraints-Based Elementary Flux Modes: Application to Escherichia coli Core Metabolism. *Processes*, 8(12).

Nieuwenhuis, R.; Oliveras, A.; and Tinelli, C. 2006. Solving SAT and SAT modulo theories: From an abstract Davis–Putnam–Logemann–Loveland procedure to DPLL (T). *Journal of the ACM (JACM)*, 53(6): 937–977.

Orth, J. D.; Thiele, I.; and Palsson, B. Ø. 2010. What is flux balance analysis? *Nature biotechnology*, 28(3): 245–248.

Reynolds, A.; King, T.; and Kuncak, V. 2017. Solving quantified linear arithmetic by counterexample-guided instantiation. *Formal Methods in System Design*, 51: 500–532.

Thuillier, K.; Baroukh, C.; Bockmayr, A.; Cottret, L.; Paulevé, L.; and Siegel, A. 2022. MERRIN: MEtabolic regulation rule INference from time series data. *Bioinformatics*, 38(Supplement_2): ii127–ii133.

Thuillier, K.; Siegel, A.; and Paulevé, L. 2023. CEGAR-based approach for solving combinatorial optimization modulo quantified linear arithmetics problems – Code and Appendix. https://doi.org/10.5281/zenodo.10361533.

Videla, S.; Saez-Rodriguez, J.; Guziolowski, C.; and Siegel, A. 2017. caspo: a toolbox for automated reasoning on the response of logical signaling networks families. *Bioinformatics*, 33(6): 947–950.

von Kamp, A.; and Klamt, S. 2014. Enumeration of Smallest Intervention Strategies in Genome-Scale Metabolic Networks. *PLOS Computational Biology*, 10(1): 1–13.

CEGAR-based approach for solving combinatorial optimization modulo quantified linear arithmetics problems — Technical Appendix —

Kerian Thuillier[1], **Anne Siegel**[1], **Loïc Paulevé**[2]

[1] Univ. Rennes, Inria, CNRS, IRISA, UMR6074, F-35000 Rennes, France
[2] Univ. Bordeaux, Bordeaux INP, CNRS, LaBRI, UMR5800, F-33400 Talence, France
kerian.thuillier@irisa.fr, anne.siegel@irisa.fr, loic.pauleve@labri.fr

Theorem 1 ($\phi \Rightarrow \phi_{\text{approx}}$)**.** *Let ϕ a* SAT+qLP *problem and ϕ_{approx} its Boolean abstraction. For any model $(x, y) \in \mathbb{B}^n \times \mathbb{R}^m$ of ϕ, there exists $\bar{f} \in \mathbb{B}^{|D|+|E|+|H|}$ such that $(x, \bar{f})$ is a model of ϕ_{approx}.*

Proof. Let $(x, y) \models \phi$ and $\bar{f}$ such that: $\forall c_h \in D \cup E \cup H, \bar{f}_{c_h} = \top \iff x \not\models c_h$. For $(x, \bar{f}) \models \phi_{\text{approx}}$, $(x, \bar{f})$ should satisfy the Eqs. 4.

(4a) As Eq. 1a equals Eq. 4a and $x \models \bigwedge_{c\in C} c(x)$, we have $(x, \bar{f}) \models \bigwedge_{c\in C} c(x)$.

(4b) By definition of $\bar{f}$, $\bar{f}_d = \top$ for each clause $d \in D$ not satisfied by x. Thus, each clause $d \in D$ is satisfied by either x or $\bar{f}$. Therefore, $(x, \bar{f}) \models \bigwedge_{d\in D} \bar{d}(x, \bar{f})$.

(4c) Using same reasoning as for (4b), there are $(x, \bar{f}) \models \bigwedge_{e\in E} \bar{e}(x, \bar{f})$ and $(x, \bar{f}) \models \bigwedge_{h\in H} \bar{h}(x, \bar{f})$. Therefore, $(x, \bar{f}) \models \bigwedge_{e\in E} \bar{e}(x, \bar{f}) \wedge \bigwedge_{h\in H} \bar{h}(x, \bar{f})$.

Therefore $(x, \bar{f}) \models \phi_{\text{approx}}$, and $\phi \implies \phi_{\text{approx}}$.

Models of (g, ϕ) are subsets of models of ϕ and by definition $\not\models (g, \phi)$ if $\not\models \phi$. Hence, $(g, \phi) \iff \phi$, i.e., $(g, \phi) \implies \phi_{\text{approx}}$.

Let $\nu^* = (x, y)$. Suppose that $\nu^* \models (g, \phi)$ with $g(\nu^*)$ its optimal value. By previous statements, $\exists \bar{f}, (x, \bar{f}) \models \phi_{\text{approx}}$. As $g : \mathbb{B}^n \to \mathbb{R}$, then $g((x, \bar{f})) = g(x) = g((x, y))$. □

Corollary 1.1. $(g, \phi) \implies \phi_{approx}$.

Proof. Models of (g, ϕ) are subsets of models of ϕ and by definition $\not\models (g, \phi)$ if $\not\models \phi$. Hence, $(g, \phi) \iff \phi$. Therefore by Theorem 1, $(g, \phi) \implies \phi_{\text{approx}}$. □

Lemma A. *Given C_h a set of hybrid clauses and $x \in \mathbb{B}^n$ a Boolean variables assignment, $y \models \mathcal{C}_x^{C_h} \iff (x, y) \models \bigwedge_{c_h\in C_h} c_h(x, y)$.*

Proof. ($\rightarrow$) Let $y \in \mathbb{R}^m$ such that $y \models \mathcal{C}_x^{C_h}$. By *reductio ad absurdum*, suppose that $\exists c_h \in C_h, (x, y) \not\models \bigwedge_{c_h\in C_h} c_h(x, y)$. The hybrid constraint $c_h(x, y)$ is of the form $\bigwedge_i x_i \bigwedge \neg x_j \wedge f_{c_h}(y) \leq 0$. Thus, there are $x \not\models \bigwedge_i x_i \bigwedge \neg x_j$ and $y \not\models f_{c_h}(y) \leq 0$. By definition of $\mathcal{C}_x^{C_h}$, if $x \not\models \bigwedge_i x_i \bigwedge \neg x_j$ then $f_{c_h}(y) \leq 0 \in \mathcal{C}_x^{C_h}$. As $y \models \mathcal{C}_x^{C_h}$, then $y \models f(y) \leq 0$. Otherwise, $x \models \bigwedge_i x_i \bigwedge \neg x_j$. This contradicts the hypothesis that $\exists c_h \in C_h, (x, y) \not\models \bigwedge_{c_h\in C_h} c_h(x, y)$. Therefore, $(x, y) \models \bigwedge_{c_h\in C_h} c_h(x, y)$.
($\leftarrow$) Let $(x, y) \models \bigwedge_{c_h\in C_h} c_h(x, y)$. Thus, $\forall c_h \in C_h$ either $x \models \bigwedge_i x_i \bigwedge_j \neg x_j$ or $y \models f_{c_h}(y) \leq 0$. Therefore, by definition of $\mathcal{C}_x^{C_h}$, $y \models \mathcal{C}_x^{C_h}$. □

Lemma B. *Given C_h a set of hybrid clauses, $\hat{c}_h$ a hybrid clause and $x \in \mathbb{B}^n$, $\not\models \mathcal{C}_x^{C_h} \vee f^*_{\hat{c}_h}(\mathcal{C}_x^{C_h}) \leq 0$ if and only if $x \models \forall y \in \mathbb{R}^m, \bigwedge_{c_h\in C_h} c_h(x, y) \implies \hat{c}_h(x, y)$.*

Proof. ($\rightarrow$) If $\not\models \mathcal{C}_x^{C_h}$, then $\forall y \in \mathbb{R}^m, (x, y) \not\models \bigwedge_{c_h\in C_h} c_h(x, y)$. Therefore, $x \models \forall y \in \mathbb{R}^m, \bigwedge_{c_h\in C_h} c_h(x, y) \implies \hat{c}_h(x, y)$. Otherwise, $\exists y \in \mathbb{R}^m, y \models \mathcal{C}_x^{C_h}$ and $f^*_{\hat{c}_h}(\mathcal{C}_x^{C_h}) \leq 0$. Thus, $\exists y \in \mathbb{R}^m, \bigwedge_{c_h\in C_h} c_h(x, y)$. By *reductio ad absurdum*, suppose that $\exists y' \in \mathbb{R}^m, (x, y') \models \bigwedge_{c_h\in C_h} c_h(x, y')$ and $f(y') > 0$. Thus, $f^*_{\hat{c}_h}(\mathcal{C}_x^{C_h}) < f(y')$. However, by definition of $f^*_{\hat{c}_h}(\mathcal{C}_x^{C_h})$, $\forall y \in \mathbb{R}^m, y \models \mathcal{C}_x^{C_h} \implies f(y) \leq f^*_{\hat{c}_h}$. It contradicts the hypothesis that $f^*_{\hat{c}_h}(\mathcal{C}_x^{C_h}) \leq 0$. Therefore, $\not\models \mathcal{C}_x^{C_h} \vee f^*_{\hat{c}_h}(\mathcal{C}_x^{C_h}) \leq 0$ implies that $x \models \forall y \in \mathbb{R}^m, \bigwedge_{c_h\in C_h} c_h(x, y) \implies \hat{c}_h(x, y)$.
($\leftarrow$) Suppose that $x \models \forall y \in \mathbb{R}^m, \bigwedge_{c_h\in C_h} c_h(x, y) \implies \hat{c}_h(x, y)$. By *reductio ad absurdum*, suppose that $\exists y \in \mathbb{R}^m$ such that $y \models \mathcal{C}_x^{C_h}$ and $f^*_{\hat{c}_h}(\mathcal{C}_x^{C_h}) > 0$. Thus, $\exists y \in \mathbb{R}^m, (y \models \mathcal{C}_x^{C_h}) \wedge f(y) > 0$. By definition of x, $\forall y' \in \mathbb{R}^m, (y \models \mathcal{C}_x^{C_h}) \implies f(y) \leq 0$. This contradicts that $\exists y \in \mathbb{R}^m, (y \models \mathcal{C}_x^{C_h}) \wedge f_{\hat{c}_h}(y) > 0$. Therefore, $x \models \forall y \in \mathbb{R}^m, \bigwedge_{c_h\in C_h} c_h(x, y) \implies \hat{c}_h(x, y)$ implies that $\not\models \mathcal{C}_x^{C_h} \vee f^*_{\hat{c}_h}(\mathcal{C}_x^{C_h}) \leq 0$. □

Theorem 2. *Let ϕ be a* SAT+qLP *formula and ϕ_{approx} its Boolean abstraction. Given $x \in \mathbb{B}^n$ and $y \in \mathbb{R}^m$, $(x, y) \models \phi$ if and only if the following three conditions hold:* ***(C1)*** *$\exists \bar{f}, (x, \bar{f}) \models \phi_{approx}$;* ***(C2)*** *$y \models \mathcal{C}_x^D$;* ***(C3)*** *$\not\models \mathcal{C}_x^E \vee \bigwedge_{h\in\mathcal{C}_x^H} f^*_h(\mathcal{C}_x^E) \leq 0$.*

Proof. ($\rightarrow$) Suppose that $(x, y) \models \phi$. By Theorem 1, $\phi \implies \phi_{\text{approx}}$. Thus, *C1* holds. As $(x, y) \models \phi$, then $(x, y) \models \bigwedge_{d\in D} d(x, y)$. Thus, Lemma A concludes that *C2* holds. As $(x, y) \models \phi$, then $x \models \forall z \in \mathbb{R}^p, \bigwedge_{e\in E} e(x, z) \implies \bigwedge_{h\in H} h(x, z)$. Thus, $\forall h \in$

$\mathcal{C}_x^H, x \models \forall z \in \mathbb{R}^p, \bigwedge_{e \in E} e(x,z) \implies h(x,z)$. Lemma B concludes that *C3* holds. Therefore, $(x,y) \models \phi$ implies *C1*, *C2* and *C3*.
($\leftarrow$) Suppose that all three conditions hold. By *C1*, $\exists \bar{f}, (x,\bar{f}) \models \phi_{\text{approx}}$. Thus, $x \models \bigwedge_{c \in C} c(x)$ (Eq. 1a). *C2* and Lemma A concludes for Eq. 1b. *C3* and Lemma B conclude for Eq. 1c. Therefore, *C1*, *C2* and *C3* implies $(x,y) \models \phi$. □

Corollary 2.1. *Given $x \in \mathbb{B}^n$ and $y \in \mathbb{R}^m$ a real-valued variables assignment, if* ***(C1')*** $\exists \bar{f}, (x,\bar{f}) \models (g, \phi_{approx})$, ***C2*** *and* ***C3*** *hold, then* $(x,y) \models (g,\phi)$.

Proof. As $(x,\bar{f}) \models (g, \phi_{\text{approx}})$, then $(x,\bar{f}) \models \phi_{\text{approx}}$. Thus, *C1* holds. Moreover, $\forall (x', \bar{f}') \models \phi, g(x) \leq g(x')$. By Corollary 1.1, $(g,\phi) \implies \phi_{\text{approx}}$. Therefore, *C1*, *C2*, *C3* hold and x is minimal according to g. Therefore, *C1'*, *C2* and *C3* implies $(x,y) \models (g,\phi)$. □

Lemma 3. $\phi \implies \phi_{approx} \wedge \phi_r^{\exists}(x)$.

Proof. Let $\bar{f}'$ such that $\forall c_h \in D \cup E \cup H, \bar{f}'_{c_h} = \top \iff x' \not\models c_h$. By Theorem 1, we have $(x', \bar{f}') \models \phi_{\text{approx}}$. If $(x,\bar{f})$ satisfies *C2* then $\phi_r^{\exists}(x)$ does not generate new constraints. Thus, $\phi_r^{\exists}(x) \wedge \phi_{\text{approx}} = \phi_{\text{approx}}$. Otherwise, *C2* does not hold for $(x,\bar{f})$. Let $\mathcal{C}_{\text{unsat}}$ be an unsatisfiable core of $\mathcal{C}_x^D$. By *reductio ad absurdum*, suppose that $\exists (x', \bar{f}') \not\models \phi_r^{\exists}(x)$. Thus, $\forall f \in \mathcal{C}_{\text{unsat}}, \bar{f}' = \top$. By definition of $\bar{f}'$ and $\mathcal{C}_{x'}^D$, it means that $\mathcal{C}_{\text{unsat}} \subseteq \mathcal{C}_{x'}^D$. Hence, $(x', f') \models \phi_r^{\exists}(x) \wedge \phi_{\text{approx}}$. Therefore, $\phi \implies \phi_{\text{approx}} \wedge \phi_r^{\exists}(x)$. □

Property 4. *Given a linear objective function $\boldsymbol{f}$ and two linear optimization problems $(\boldsymbol{f}, \mathcal{C}_1)$ and $(\boldsymbol{f}, \mathcal{C}_2)$, $\mathcal{C}_1 \subseteq \mathcal{C}_2 \implies \boldsymbol{f}^*(\mathcal{C}_1) \geq \boldsymbol{f}^*(\mathcal{C}_2)$.*

Proof. By *reductio ad absurdum*, suppose that $\mathcal{C}_1 \subseteq \mathcal{C}_2$ and $f^*(\mathcal{C}_1) < \mathcal{C}_2$. Let $y = \text{argmax}_{y \models \mathcal{C}_2} \boldsymbol{f}(y)$. As $\mathcal{C}_1 \subseteq \mathcal{C}_2$, then $y \models \mathcal{C}_1$. Since $f(y) = f^*(\mathcal{C}_2)$ and $y \models \mathcal{C}_1$, its contradicts $f^*(\mathcal{C}_1) < \mathcal{C}_2$. Therefore, $\mathcal{C}_1 \subseteq \mathcal{C}_2 \implies f^*(\mathcal{C}_1) \geq \mathcal{C}_2$. □

Lemma 5. $\phi \implies \phi_{approx} \wedge \phi_r^{\forall}(x)$

Proof. Let $\bar{f}'$ such that $\forall c_h \in D \cup E \cup H, \bar{f}'_{c_h} = \top \iff x' \not\models c_h$. By Theorem 1, we have $(x', \bar{f}') \models \phi_{\text{approx}}$. If $(x,\bar{f})$ satisfies *C3* then $\phi_r^{\forall}(x)$ does not generate new constraints. Thus, $\phi_r^{\forall}(x) \wedge \phi_{\text{approx}} = \phi_{\text{approx}}$. Otherwise, *C3* does not hold for $(x,\bar{f})$. By *reductio ad absurdum*, suppose that $\exists (x', \bar{f}') \not\models \phi_r^{\forall}(x)$. Let $h \in H$ such that $f_h \in \mathcal{C}_x^H$ and $f_h^*(\mathcal{C}_x^E) > 0$. Such h exists as *C3* does not hold for x. By definition of $(x', y') \models \phi$ and $\bar{f}'$, there are either $\bar{f}'_h = \bot$ or $\bar{f}'_h = \top \wedge f_h^*(\mathcal{C}_{x'}^E) \leq 0$. For the first case, $\bar{f}'_h$ satisfies the constraint of $\phi_r^{\exists}(x)$ associated with h. For the second case, suppose that $\bar{f}'_h = \top \wedge f_h^*(\mathcal{C}_{x'}^E) \leq 0$. Let $\mathcal{C}_{\text{opt}}^{f_h}$ be an optimal core of $(f_h, \mathcal{C}_x^E)$. Thus, $\forall e \in E, f_e \notin \mathcal{C}_{\text{opt}}^{f_h} \implies \bar{f}'_e = \bot$ and $\bar{f}_h = \top$. By definition of $\bar{f}'$ and $\mathcal{C}_{x'}^E$, it means that $\mathcal{C}_{x'}^E \subseteq \mathcal{C}_{\text{opt}}^{f_h}$. However, we have that $f_h^*(\mathcal{C}_{x'}^E) < f_h^*(\mathcal{C}_x^E)$. This contradicts property 4. Hence, $(x', f') \models \phi^{\forall} r(x) \wedge \phi_{\text{approx}}$. Therefore, $\phi \implies \phi_{\text{approx}} \wedge \phi_r^{\forall}(x)$. □

Theorem 6. *Given $(x,\bar{f}) \models \phi_{approx}$, $\phi \implies \phi_r^{\exists}(x) \wedge \phi_r^{\forall}(x) \wedge \phi_{approx}$.*

Proof. By Lemma 3, we have $\phi \implies \phi_r^{\exists}(x) \wedge \phi_{\text{approx}}$. By Lemma 5, we have $\phi \implies \phi_r^{\forall}(x) \wedge \phi_{\text{approx}}$. As the constraints generated by $\phi_r^{\exists}(x)$ and $\phi_r^{\forall}(x)$ impact disjoint sets of variables $\bar{f}$, then $\phi \implies \phi_r^{\exists}(x) \wedge \phi_r^{\forall}(x) \wedge \phi_{\text{approx}}$. □

Corollary 6.1. $(g,\phi) \implies \phi_r^{\exists}(x) \wedge \phi_r^{\forall}(x) \wedge \phi_{approx}$.

Proof. By definition, $\phi \iff (g,\phi)$. Therefore by Theorem 6, $(g,\phi) \implies \phi_r^{\exists}(x) \wedge \phi_r^{\forall}(x) \wedge \phi_{\text{approx}}$. □

Corollary 6.2. $\forall \nu^* \models (g,\phi) \implies \exists \nu' \models \phi_r^{\exists}(x) \wedge \phi_r^{\forall}(x) \wedge \phi_{approx}, g(\nu') = g(\nu^*)$.

Proof. Suppose that $(x', y') \models (g,\phi)$ with $g((x',y'))$ its optimal value. By definition, $\exists \bar{f}', (x', \bar{f}') \models \phi_r^{\exists}(x) \wedge \phi_r^{\forall}(x) \wedge \phi_{\text{approx}}$. As $g : \mathbb{B}^n \to \mathbb{R}$, then $g((x',\bar{f}')) = g(x') = g((x',y'))$. □

Lemma 7. $\exists y \in \mathbb{R}^m, \mathcal{C}_x^D \iff \bigwedge_{\mathcal{P}_i \in \mathcal{P}_x^D} y \models \mathcal{P}_i$.

Proof. ($\rightarrow$) Suppose that $\exists y \in \mathbb{R}^m, \mathcal{C}_x^D$ and it exists $\mathcal{P}_i \in \mathcal{P}_x^D$ unsatisfiable. We know that $\mathcal{P}_x^D$ is a partition of $\mathcal{C}_x^D$, hence $\mathcal{P}_i \subseteq \mathcal{C}_x^D$. If $\mathcal{P}_i$ is unsatisfiable, so is $\mathcal{C}_x^D$. Therefore, it could not exists $\mathcal{P}_i \in \mathcal{P}_x^D$ unsatisfiable if $\mathcal{C}_x^D$ is satisfiable. ($\leftarrow$) Suppose that $\forall \mathcal{P}_i \in \mathcal{P}_x^D, y_i \models \mathcal{P}_i$. We know that $\mathcal{P}_x^D$ is a partition of $\mathcal{C}_x^D$ such that no variables are shared among the constraints of different partitions. Hence $y = y_{ii} \models \bigwedge_{\mathcal{P}_i \in \mathcal{P}_x^D} \mathcal{P}_i$, and $\bigwedge_{\mathcal{P}_i \in \mathcal{P}_x^D} y \models \mathcal{P}_i$. As y is a model of all the subsets in the partition $\mathcal{P}_x^D$, $y \models \mathcal{C}_x^D$. □

Lemma 8. *If $\mathcal{C}_x^E$ is satisfiable, then $f_h^*(\mathcal{C}_x^E) = f_h^*(\mathcal{P}'^E_x)$.*

Proof. By definition of $\mathcal{P}$, we have that all the linear constraints that can have an impact on the variables involved in f_h are in P'. Therefore, linear constraints in the other subsets will not impact the variables involved in f_h. These constraints can only impact the satisfiability of the problem, however, we supposed that $\mathcal{C}_x^E$ is satisfiable. Hence, the optimum of $(f_h, \mathcal{C}_x^E)$ depends only of the constraints in $\mathcal{P}'^E_x$. □

Conclusion and Perspectives

In this chapter

Conclusion

The objective of this thesis was to define, formalize, and implement new methods to infer Boolean networks (BNs) controlling the metabolism, including the metabolic feedback and control rules that are not inferred with state-of-the-art methods. The works presented in this manuscript aim to address this objective. Our contributions fall into three areas: *bioinformatics*, *formal methods*, and *resources* made available to the community.

Contributions in Bioinformatics

When we started this thesis, we intuited that it would be necessary to integrate the linear dynamics of the metabolism into the definition of the inference problem to infer feedback and control rules. This led us to **define inference problem considering the hybrid dynamics of regulated metabolic networks** (**Chapter II**), whereas state-of-the-art methods only consider the discrete dynamics of the regulatory network. In particular, we relied on the regulated flux balance analysis (rFBA) framework to model the coupled dynamics of the regulatory and metabolic networks.

From this definition, we derived **three formulations of the inference problem**, of which two are equivalent, and two dedicated methods to address them:

— The first method (**Chapter III**) uses an Answer Set Programming (ASP) encoding to solve a **relaxed abstraction of the inference problem**, formulated

as a Boolean satisfiability problem with quantifiers. This relaxed abstraction is based on a discrete over-approximation of the rFBA dynamics, that discretized metabolic steady-states (the FBA equations).

— The second method (**Chapter IV**) addresses the **hybrid formulation of the inference problem**. To solve it, we have developed a **dedicated hybrid solving workflow** that integrates the FBA equations with ASP.

These methods have been validated and tested through case studies and benchmarks using regulated metabolic networks of *Escherichia coli*. In particular, we developed a time series generation protocol to generate synthetic time series of omics data from rFBA simulations. Using this protocol, we generate a comprehensive benchmark to validate our inferring methods. The results for the second method (*MERRIN*'s workflow) demonstrate that the **inferring of regulatory rules, including metabolic feedback and control, is possible solely from time series of kinetics and transcriptomics data**. These two data types are commonly used to study the behaviors of bacteria.

Contributions in Formal Methods

Retrospectively, we did not expect that the solving of the inference problems would become a limiting factor. We initially assumed that the primary challenge would be to formally define and validate the inference problem, not to solve it. Indeed, modern SAT and Satisfiability Modulo Theory (SMT) solvers are widely used in the industry to efficiently solve hybrid problems. The inability of these solvers to efficiently solve our inference problem highlighted the fact that **problems from systems biology have specificities and solving requirements** not encountered in industry's problems.

Specificities of problems from systems biology. In systems biology, problems are characterized by complex combinatorial search spaces constrained by biological knowledge, and optimization functions designed to accommodate experimental noise in input data. Solving these problems often requires enumerating all the solutions, or at least sampling the solution space, since it is impossible to discriminate between optimal solutions. They are all equally supported by the biological knowledge defining the search space. Therefore, it is necessary to enumerate solutions to gain a comprehensive understanding of the solution space, deduce new knowledge, identify bottlenecks in current knowledge, and plan new experimental protocols. There is thus a **need to define novel solving formalisms and tools adapted to the specificities of systems biology problems**.

To overcome this bottleneck and enable solving our hybrid definition of the inference problem, we **developed new formal methods to enumerate solutions**

of hybrid combinatorial optimization problems. In **Chapter IV**, we present ***MERRIN*'s workflow** for inferring Boolean regulatory rules controlling the metabolism. In **Chapter V**, we take this further by presenting a **generic solving method** that allows enumerating solutions of combinatorial optimization problems with quantified linear constraints, a class of problems of which the hybrid inference problem belongs.

Contributed Resources

Throughout this thesis, we have **prioritized the reproducibility** of our work and results. To this end, along with the theoretical contributions presented in this manuscript, we have made available all the regulated metabolic networks (RMNs), our tools, and our implementations in public repositories.

Regulated metabolic network models. Finding suitable RMNs for benchmarking and validating our methods has been quite challenging. There are only a **few RMNs available in the literature**, and even fewer are associated with experimental conditions and predictions of cell behaviors that can be used. When presenting the work of this thesis, many were surprised to discover that such models exist and are publicly available.

We only found three RMNs that could be used: a model of core-carbon metabolism (Covert et al., 2001), a medium-scale model of *Escherichia coli*'s core metabolism (Covert and Palsson, 2002), and a large-scale model of *Escherichia coli* (Covert et al., 2004). Except for the large-scale model, we use these RMNs to validate and benchmark our methods.

A significant portion of this thesis has been dedicated to cleaning and updating these three RMNs. Indeed, the Boolean regulatory rules of **these models were only described in 'textual' form**, either in some supplementary materials or in some *Excel sheets*, **necessitating extensive preprocessing** to be usable with modern simulation tools, such as *FlexFlux* (Marmiesse et al., 2015). This preprocessing includes:

1. standardizing notations for genes, proteins, and metabolites within rules of the same model;

2. converting (and debugging) the regulatory rules from their textual description to a standard syntax of Boolean functions;

3. validating the cleaned and updated networks according to the model dynamics under various experimental conditions described in the model's introductory paper.

This tedious task has been manually performed for the three RMNs. At the end of the process, **we produced standard files describing each RMN**, with the metabolic network in the format *SBML* (Hucka et al., 2003), and the regulatory network in the format *SBML-qual* (Chaouiya et al., 2013). These files are available at https://github.com/kthuillier/regulated-metabolic-models. Comprehensive descriptions of the core-carbon metabolism and medium-scale models, as well as our validation protocols, are provided in Appendix A.

Tools and datasets. These RMNs have been **used to generate synthetic time series data for validating and benchmarking** the different methods presented in this manuscript. The relaxed inference problem, introduced in chapter III, has been validated on a *toy* RMN derived from the model of core-carbon metabolism (https://github.com/bioasp/boolean-caspo-flux).

In chapter IV, we present *MERRIN*[8] a dedicated tool to infer Boolean regulatory rules controlling metabolic networks from omics time series data. *MERRIN* has been **benchmarked on a synthetic dataset based on simulated time series of different complexity and combinations of omics data**. The benchmark generation protocol and scripts required to solve it with *MERRIN* are available at https://github.com/bioasp/merrin-covert.

In particular, we rely on this benchmark generation protocol to validate *MerrinASP*[9], a generic hybrid solver used to address OPT+qLP problems introduced in chapter V. To our knowledge, **no benchmarks for OPT+qLP problems were previously available** in the literature, even though other solvers can theoretically address this class of problems. Therefore, we introduced our **own datasets based on instances of the inference problem** to benchmark *MerrinASP*. They are available at https://zenodo.org/records/10361533.
We expect that these datasets will find applications beyond the field of systems biology, particularly in the operational research domain. Specifically, we hope they will facilitate the development of novel solving methods adapted to the specificities of problems from systems biology.

[8] *MERRIN* is available at https://github.com/bioasp/merrin.
[9] *MerrinASP* is available at https://github.com/kthuillier/merrinasp.

Perspectives

Following the work presented in this manuscript, my future research projects would aim to expand the scope and enhance the biological accuracy of our inference methods. Specifically, three perspectives, ranging from short-term to long-term, can be proposed to reach these objectives. These perspectives differ in the domain they involve: **(i)** extending the inference methods' scope (*short-term*); **(ii)** more accurate modeling of regulated metabolic network (RMN) dynamics (*medium to long-term*); and **(iii)** novel hybrid formal methods (*long-term*).

Maintenance and Updating of Regulated Metabolic Networks

In this manuscript, we define inferring methods that infer Boolean regulatory rules supported by interactions. However, it is **not always relevant to infer all regulatory rules** from scratch. Some Boolean regulatory rules are already known and verified, and large-scale Boolean regulatory networks have even been reconstructed and curated. It would be a mistake not to exploit these resources and knowledge in our inference methods.

RMN update methods. Following on from *MERRIN*'s inferring workflow, a **short-term perspective would be to develop methods for automatically updating Boolean regulatory networks**. The development of such update methods will accelerate the emergence of high-quality RMNs by enabling the maintenance and updating of existing models with new datasets. In particular, it could allow **adapting Boolean networks reconstructed without metabolic feedbacks and controls** by only needing to infer these specific rules.

An update tool would take as input an RMN, a set of interactions, and new omics data. Its objective would be to determine whether the RMN is compatible with the omics data, and if not, to identify and correct a minimal set of regulatory rules for the RMN to become compatible with the new data. Integrating already known Boolean rules into the inferring process is already handled in some Boolean network inferring tools, like *BoNesis* (Chevalier et al., 2020). In practice, it could be easily integrated into our inference methods by profiting from our generic *MerrinASP*'s encoding of the inference problem.

Challenges. The primary challenge would concern the guarantees that should be given to the updating process. Specifically, it must be decided whether the updated model should remain fully compatible with all the data used in its reconstruction, or be solely adapted to new data. In the former case, the number of observations with which the model must remain compatible will increase with each update,

making the updating process progressively more complex. This raises the issue of defining optimal data representation and identifying the minimal set of observations describing the model.

Application to reproducibility. In addition to facilitating the development of new models of RMNs, such tools could be used for **reproducibility purposes**. During this thesis, I reconstructed a large-scale regulatory network of *Escherichia coli* based on the regulatory rules described in the appendices of Covert et al. (2004). However, the reconstructed model does not fully reproduce all the results presented in the aforementioned paper. In this context, update methods could be used to adapt, or correct, the RMN to reproduce the paper's results.

More Accurate Modeling of RMN Dynamics

Missing regulatory rules. In chapter IV, we demonstrated that our inference method, *MERRIN*, infers Boolean networks that are smaller than the ground truth networks, meaning that **not all regulatory rules of the ground-truth model are needed to explain the input observations**. The non-inferred rules do not impact the rFBA dynamics of RMNs. To infer these missing rules, **additional information such as enzyme concentrations would be necessary**.

Threshold-based rules. Moreover, in this thesis, we have **not accounted for the impact of metabolite concentrations and reaction activity rates** on the regulatory system. We only consider the availability of environmental metabolites and the state (active or inactive) of reactions, which does not accurately reflect the biological functioning of cells. In reality, cells possess sensors that estimate the abundance of metabolites (both environmental and intracellular). Consequently, certain regulatory mechanisms are activated only when metabolite concentrations reach specific thresholds. Formally, this **requires defining Boolean rules with threshold conditions** on metabolite concentrations.

Advanced RMN Dynamics. A **medium to long-term perspective would be to reformulate the inference problem using RMN dynamics accounting for enzyme production and degradation, as well as threshold-based regulatory rules**. For instance, the de-rFBA formalism (Liu and Bockmayr, 2020) could be used.

Addressing the inference problem with such dynamics will introduce new challenges in both the solving methods and the types of data required for inference. Solving this problem will likely necessitate developing new hybrid solving frameworks, in particular, based on hybrid automaton inference methods. The de-rFBA

formalism models the RMN as a hybrid automaton. Regarding experimental data types, it would be interesting to assess whether the trade-off between the quantity and complexity of the data needed for the inference and the quality of the inferred networks is cost-effective compared to what *MERRIN* infers from kinetic and transcriptomic data.

Inference of Missing Interactions Using Machine Learning and Formal Methods

A **significant limitation** of the inference methods presented in this manuscript is the need **for comprehensive prior knowledge** of all interactions between genes, proteins, metabolites, and reactions involved in regulation. Our methods can only infer regulatory rules supported by the provided interactions. If interactions are missing, our methods, *MERRIN*, may fail to infer any regulatory rules.

Use of interaction databases. In practice, it is possible to use interactions available in gene regulatory network (GRN) databases. However, these databases have limitations: they often lack the interactions between the metabolic and regulatory scales, and many of the interactions they contain are statistically inferred and do not have guarantees to be biologically relevant. Therefore, using interactions from these databases necessitates carefully selecting the relevant interactions to use for the inferring. This solution is not ideal as it requires manual curation of database interactions and does not address the lack of interactions between the metabolic and regulatory scales.

Coupling machine learning and formal methods. To address this issue, a **long-term perspective would be to infer the missing interactions directly during the inference process** using statistical or machine learning-based methods (Gao et al., 2020; Liu et al., 2021; Barman and Kwon, 2020). The primary drawback of these methods is their tendency to infer spurious gene interactions, that is, interactions lacking biological relevance. They do not provide the guarantees and explainability provided by formal methods, such as logic programming. For instance, our Boolean network inference methods can explain why each regulatory rule has been inferred. By prohibiting a rule to be learned, one can identify which logic rules, and thus observations, are violated. Such explainability is most of the time not possible with machine learning-based approaches.

Recent studies have explored the integration of logic programming and machine learning methods to infer biologically relevant Boolean networks within the context of drug development (Réda and Delahaye-Duriez, 2022). To achieve that, the authors introduce an inferring workflow that first infers candidate Boolean networks

with an ASP-based inferring method; and then filters them based on an 'influence maximization' criterion commonly used in machine learning.

Similar methods could surely be developed to integrate the selection and inferring of new relevant interactions into our inferring process. The **integration of machine learning and formal methods is a fast-growing field of research** with the potential to open up new opportunities for the inference of Boolean regulatory networks.

Conclusion. The work conducted during this thesis focused on developing inference methods for Boolean regulatory rules that control metabolic processes. These methods are based on constraint programming formalisms and reasoning from biological knowledge. These reasonings focus on the compatibility between biological data and the assumed multi-scale dynamics of biological systems.

In the long term, the objective would be to achieve this inference using machine learning-based approaches. Currently, the limited availability of RMNs in the literature prevents the training of such models.

A first strategy to increase the training set would be to develop models in collaboration with biologists that, through predictions, would facilitate the design of experimental protocols. These new experiments will enable the acquisition of new data which will be used for the development and enhancement of RMNs.

A second strategy would be to refine the modeling, simulation, and inference formalisms of RMNs to gain a better understanding of the biological mechanisms of metabolic regulation.

By integrating formal methods with machine learning, improving modeling formalisms, and developing decision-aid protocols, the reconstruction of RMN models can be simplified, leading to a better understanding of the regulatory rules controlling cell behaviors.

List of Figures

List of Tables

Bibliography

Allart, E., Niehren, J., and Versari, C. (2021). Exact boolean abstraction of linear equation systems. *Computation*, 9(11).

Badia-i Mompel, P., Wessels, L., Müller-Dott, S., Trimbour, R., Ramirez Flores, R. O., Argelaguet, R., and Saez-Rodriguez, J. (2023). Gene regulatory network inference in the era of single-cell multi-omics. *Nature Reviews Genetics*, 24(11):739–754.

Banbara, M., Kaufmann, B., Ostrowski, M., and Schaub, T. (2017). Clingcon: The next generation. *TPLP*, 17(4):408–461.

Baral, C. (2003). *Knowledge Representation, Reasoning and Declarative Problem Solving*. Cambridge University Press, New York, NY, USA.

Barman, S. and Kwon, Y.-K. (2020). A neuro-evolution approach to infer a Boolean network from time-series gene expressions. *Bioinformatics*, 36(Supplement_2):i762–i769.

Barrett, C. and Tinelli, C. (2018). *Satisfiability modulo theories*. Springer.

Becker-Hapak, M. and Eisenstark, A. (1995). Role of rpoS in the regulation of glutathione oxidoreductase (gor) in Escherichia coli. *FEMS Microbiology Letters*, 134(1):39–44.

Bernot, G., Comet, J.-P., Richard, A., and Guespin, J. (2004). Application of formal methods to biological regulatory networks: extending thomas' asynchronous logical approach with temporal logic. *Journal of Theoretical Biology*, 229(3):339–347.

Bornstein, B. J., Keating, S. M., Jouraku, A., and Hucka, M. (2008). Libsbml: an api library for sbml. *Bioinformatics*, 24(6):880–881.

Bourqui, R., Cottret, L., Lacroix, V., Auber, D., Mary, P., Sagot, M.-F., and Jourdan, F. (2007). Metabolic network visualization eliminating node redundance and preserving metabolic pathways. *BMC systems biology*, 1:1–19.

Brummayer, R. and Biere, A. (2008). Lemmas on demand for the extensional theory of arrays. In *Proceedings of the Joint Workshops of the 6th International Workshop on Satisfiability Modulo Theories and 1st International Workshop on Bit-Precise Reasoning*, SMT '08/BPR '08, page 6–11, New York, NY, USA. Association for Computing Machinery.

Brummayer, R. and Biere, A. (2009). Effective bit-width and under-approximation. In *International Conference on Computer Aided Systems Theory*, pages 304–311. Springer.

Buescher, J. M., Liebermeister, W., Jules, M., Uhr, M., Muntel, J., Botella, E., Hessling, B., Kleijn, R. J., Chat, L. L., Lecointe, F., Mader, U., Nicolas, P., Piersma, S., Rugheimer, F., Becher, D., Bessieres, P., Bidnenko, E., Denham, E. L., Dervyn, E., Devine, K. M., Doherty, G., Drulhe, S., Felicori, L., Fogg, M. J., Goelzer, A., Hansen, A., Harwood, C. R., Hecker, M., Hubner, S., Hultschig, C., Jarmer, H., Klipp, E., Leduc, A., Lewis, P., Molina, F., Noirot, P., Peres, S., Pigeonneau, N., Pohl, S., Rasmussen, S., Rinn, B., Schaffer, M., Schnidder, J., Schwikowski, B., Dijl, J. M. V., Veiga, P., Walsh, S., Wilkinson, A. J., Stelling, J., Aymerich, S., and Sauer, U. (2012). Global network reorganization during dynamic adaptations of bacillus subtilis metabolism. *Science*, 335(6072):1099–1103.

Carthew, R. W. (2021). Gene regulation and cellular metabolism: an essential partnership. *Trends in Genetics*, 37(4):389–400.

Chandrasekaran, S. and Price, N. D. (2010). Probabilistic integrative modeling of genome-scale metabolic and regulatory networks in escherichia coli and mycobacterium tuberculosis. *Proceedings of the National Academy of Sciences*, 107(41):17845–17850.

Chaouiya, C., Bérenguier, D., Keating, S. M., Naldi, A., Van Iersel, M. P., Rodriguez, N., Dräger, A., Büchel, F., Cokelaer, T., Kowal, B., et al. (2013). Sbml qualitative models: a model representation format and infrastructure to foster interactions between qualitative modelling formalisms and tools. *BMC systems biology*, 7:1–15.

Chaves, M., Oyarzún, D. A., and Gouzé, J.-L. (2019). Analysis of a genetic-metabolic oscillator with piecewise linear models. *Journal of Theoretical Biology*, 462:259–269.

Chaves, M., Tournier, L., and Gouzé, J.-L. (2010). Comparing boolean and piecewise affine differential models for genetic networks. *Acta Biotheor*, 58(2-3):217–232.

Chevalier, S., Froidevaux, C., Paulevé, L., and Zinovyev, A. (2019). Synthesis of boolean networks from biological dynamical constraints using answer-set programming. In *2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 34–41. IEEE.

Chevalier, S., Noël, V., Calzone, L., Zinovyev, A., and Paulevé, L. (2020). Synthesis and simulation of ensembles of boolean networks for cell fate decision. In Abate,

A., Petrov, T., and Wolf, V., editors, *Computational Methods in Systems Biology*, pages 193–209, Cham. Springer International Publishing.

Cimatti, A., Griggio, A., and Sebastiani, R. (2011). Computing small unsatisfiable cores in satisfiability modulo theories. *Journal of Artificial Intelligence Research*, 40:701–728.

Clarke, E., Grumberg, O., Jha, S., Lu, Y., and Veith, H. (2003). Counterexample-guided abstraction refinement for symbolic model checking. *Journal of the ACM (JACM)*, 50(5):752–794.

Cottret, L. and Jourdan, F. (2010). Graph methods for the investigation of metabolic networks in parasitology. *Parasitology*, 137(9):1393–1407.

Covert, M. and Palsson, B. (2002). Transcriptional regulation in constraints-based metabolic models of escherichia coli. *The Journal of biological chemistry*, 277:28058–64.

Covert, M. W., Knight, E. M., Reed, J. L., Herrgard, M. J., and Palsson, B. O. (2004). Integrating high-throughput and computational data elucidates bacterial networks. *Nature*, 429(6987):92–96.

Covert, M. W., Schilling, C. H., and Palsson, B. (2001). Regulation of gene expression in flux balance models of metabolism. *Journal of Theoretical Biology*, 213(1):73–88.

Covert, M. W., Xiao, N., Chen, T. J., and Karr, J. R. (2008). Integrating metabolic, transcriptional regulatory and signal transduction models in escherichia coli. *Bioinformatics*, 24(18):2044–2050.

de Jong, H. (2002). Modeling and simulation of genetic regulatory systems: A literature review. *Journal of Computational Biology*, 9:67–103.

De Moura, L. and Bjørner, N. (2008). Z3: An efficient smt solver. In *International conference on Tools and Algorithms for the Construction and Analysis of Systems*, pages 337–340. Springer.

Dutertre, B. and De Moura, L. (2006). Integrating simplex with dpll (t). *Computer Science Laboratory, SRI International, Tech. Rep. SRI-CSL-06-01*.

Edwards, J. S. and Palsson, B. O. (1999). Systems properties of the haemophilus influenzae metabolic genotype. *Journal of Biological Chemistry*, 274(25):17410–17416.

Eiter, T. and Gottlob, G. (1995). On the computational cost of disjunctive logic programming: Propositional case. *Annals of Mathematics and Artificial Intelligence*, 15(3-4):289–323.

Eiter, T., Ianni, G., and Krennwallner, T. (2009). *Answer Set Programming: A Primer*, pages 40–110. Springer Berlin Heidelberg, Berlin, Heidelberg.

Farzan, A. and Kincaid, Z. (2016). Linear arithmetic satisfiability via strategy improvement. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*.

Feist, A. M. and Palsson, B. O. (2010). The biomass objective function. *Curr Opin Microbiol*, 13(3):344–349.

Frioux, C., Schaub, T., Schellhorn, S., Siegel, A., and Wanko, P. (2019). Hybrid metabolic network completion. *Theory and Practice of Logic Programming*, 19(1):83–108.

Gao, S., Sun, C., Xiang, C., Qin, K., and Lee, T. H. (2020). Learning asynchronous boolean networks from single-cell data using multiobjective cooperative genetic programming. *IEEE Transactions on Cybernetics*, 52(5):2916–2930.

Gebser, M., Janhunen, T., and Rintanen, J. (2014). Answer set programming as sat modulo acyclicity. In *ECAI*, volume 263, pages 351–356.

Gebser, M., Kaminski, R., Kaufmann, B., and Schaub, T. (2012). *Answer Set Solving in Practice*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan and Claypool Publishers.

Gebser, M., Kaminski, R., Kaufmann, B., and Schaub, T. (2017). Multi-shot ASP solving with clingo. *CoRR*, abs/1705.09811.

Gebser, M., Kaminski, R., and Schaub, T. (2011). Complex optimization in answer set programming. *Theory and Practice of Logic Programming*, 11(4-5):821–839.

Gebser, M., Kaufmann, B., Romero, J., Otero, R., Schaub, T., and Wanko, P. (2013). Domain-specific heuristics in answer set programming. *Proceedings of the AAAI Conference on Artificial Intelligence*, 27(1).

Gelfond, M. and Lifschitz, V. (1988). The stable model semantics for logic programming. In Kowalski, R., Bowen, and Kenneth, editors, *Proceedings of International Logic Programming Conference and Symposium*, pages 1070–1080. MIT Press.

Goelzer, A. and Fromion, V. (2011). Bacterial growth rate reflects a bottleneck in resource allocation. *Biochimica et Biophysica Acta (BBA) - General Subjects*, 1810(10):978–988. Systems Biology of Microorganisms.

Goelzer, A., Muntel, J., Chubukov, V., Jules, M., Prestel, E., Nölker, R., Mariadassou, M., Aymerich, S., Hecker, M., Noirot, P., Becher, D., and Fromion, V. (2015). Quantitative prediction of genome-wide resource allocation in bacteria. *Metabolic Engineering*, 32:232–243.

Gu, C., Kim, G. B., Kim, W. J., Kim, H. U., and Lee, S. Y. (2019). Current status and applications of genome-scale metabolic models. *Genome biology*, 20:1–18.

Hengge, R. (1996). Regulation of gene expression during entry into stationary phase. *Escherichia coli and Salmonella Typhimurium*, pages 1497–1512.

Hucka, M., Finney, A., Sauro, H. M., Bolouri, H., Doyle, J. C., Kitano, H., Arkin, A. P., Bornstein, B. J., Bray, D., Cornish-Bowden, A., et al. (2003). The systems biology markup language (sbml): a medium for representation and exchange of biochemical network models. *Bioinformatics*, 19(4):524–531.

Janhunen, T., Kaminski, R., Ostrowski, M., Schellhorn, S., Wanko, P., and Schaub, T. (2017). Clingo goes linear constraints over reals and integers. *Theory and Practice of Logic Programming*, 17(5-6):872–888.

Janota, M., Klieber, W., Marques-Silva, J., and Clarke, E. (2016). Solving qbf with counterexample guided refinement. *Artificial Intelligence*, 234:1–25.

Jeanne, G., Goelzer, A., Tebbani, S., Dumur, D., and Fromion, V. (2018). Dynamical resource allocation models for bioreactor optimization. *IFAC-PapersOnLine*, 51(19):20–23.

Joyce, A. R. and Palsson, B. Ø. (2006). The model organism as a system: integrating'omics' data sets. *Nature reviews Molecular cell biology*, 7(3):198–210.

Julien-Laferrière, A., Bulteau, L., Parrot, D., Marchetti-Spaccamela, A., Stougie, L., Vinga, S., Mary, A., and Sagot, M.-F. (2016). A combinatorial algorithm for microbial consortia synthetic design. *Scientific Reports*, 6(1):29182.

Jung, I. L. and Kim, I. G. (2003). Transcription of ahpc, katg, and kate genes in escherichia coli is regulated by polyamines: polyamine-deficient mutant sensitive to h2o2-induced oxidative damage. *Biochemical and Biophysical Research Communications*, 301(4):915–922.

Kaminksi, R., Romero, J., Schaub, T., and Wanko, P. (2023). How to build your own asp-based system?! *Theory and Practice of Logic Programming*, 23(1):299–361.

Kanehisa, M. and Goto, S. (2000). KEGG: Kyoto Encyclopedia of Genes and Genomes. *Nucleic Acids Research*, 28(1):27–30.

Kauffman, S. A. (1969). Metabolic stability and epigenesis in randomly constructed genetic nets. *Journal of theoretical biology*, 22(3):437–467.

Kaufmann, B., Leone, N., Perri, S., and Schaub, T. (2016). Grounding and solving in answer set programming. *AI magazine*, 37(3):25–32.

Khasidashvili, Z., Korovin, K., and Tsarkov, D. (2015). Epr-based k-induction with counterexample guided abstraction refinement. In *GCAI*, pages 137–150.

King, Z. A., Lu, J., Dräger, A., Miller, P., Federowicz, S., Lerman, J. A., Ebrahim, A., Palsson, B. O., and Lewis, N. E. (2015). BiGG Models: A platform for integrating, standardizing and sharing genome-scale models. *Nucleic Acids Research*, 44(D1):D515–D522.

Kitano, H. (2002a). Computational systems biology. *Nature*, 420(6912):206–210.

Kitano, H. (2002b). Systems biology: a brief overview. *science*, 295(5560):1662–1664.

Lagniez, J.-M., Berre, D. L., de Lima, T., and Montmirail, V. (2017). A recursive shortcut for cegar: Application to the modal logic k satisfiability problem. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pages 674–680.

Liu, L. and Bockmayr, A. (2020). Regulatory dynamic enzyme-cost flux balance analysis: A unifying framework for constraint-based modeling. *Journal of Theoretical Biology*, 501:110317.

Liu, X., Wang, Y., Shi, N., Ji, Z., and He, S. (2021). Gapore: Boolean network inference using a genetic algorithm with novel polynomial representation and encoding scheme. *Knowledge-Based Systems*, 228:107277.

Lobo, J., Minker, J., and Rajasekar, A. (1992). *Foundations of disjunctive logic programming*. MIT press.

Lougee-Heimer, R. (2003). The common optimization interface for operations research: Promoting open-source software in the operations research community. *IBM Journal of Research and Development*, 47(1):57–66.

Mahadevan, R., Edwards, J. S., and Doyle, F. J. (2002). Dynamic flux balance analysis of diauxic growth in escherichia coli. *Biophysical Journal*, 83(3):1331–1340.

Mahout, M., Carlson, R. P., and Peres, S. (2020). Answer set programming for computing constraints-based elementary flux modes: Application to escherichia coli core metabolism. *Processes*, 8(12).

Marmiesse, L., Peyraud, R., and Cottret, L. (2015). Flexflux: combining metabolic flux and regulatory network analyses. *BMC systems biology*, 9(1):1–13.

Mata, A. M., Carmen, M., and López-Barea, J. (1984). Purification by affinity chromatography of glutathione reductase (ec 1.6.4.2) from escherichia coli and characterization of such enzyme. *Zeitschrift für Naturforschung C*, 39(9-10):908–915.

Min Lee, J., Gianchandani, E. P., Eddy, J. A., and Papin, J. A. (2008). Dynamic analysis of integrated signaling, metabolic, and regulatory networks. *PLoS computational biology*, 4(5):e1000086.

Monk, J. M., Lloyd, C. J., Brunk, E., Mih, N., Sastry, A., King, Z., Takeuchi, R., Nomura, W., Zhang, Z., Mori, H., et al. (2017). i ml1515, a knowledgebase that computes escherichia coli traits. *Nature biotechnology*, 35(10):904–908.

Monod, J. (1942). Recherches sur la croissance des cultures bacteriennes. *Ann. Inst. Pasteur*, 69:179.

Moulin, C., Tournier, L., and Peres, S. S. (2021). Combining Kinetic and Constraint-Based Modelling to Better Understand Metabolism Dynamics. *Processes*, 9(10):1701.

Müller-Dott, S., Tsirvouli, E., Vazquez, M., Ramirez Flores, R. O., Badia-i Mompel, P., Fallegger, R., Türei, D., Lægreid, A., and Saez-Rodriguez, J. (2023). Expanding the coverage of regulons from high-confidence prior knowledge for accurate estimation of transcription factor activities. *Nucleic Acids Research*, 51(20):10934–10949.

Nieuwenhuis, R., Oliveras, A., and Tinelli, C. (2006). Solving sat and sat modulo theories: From an abstract davis–putnam–logemann–loveland procedure to dpll (t). *Journal of the ACM (JACM)*, 53(6):937–977.

Orth, J. D., Conrad, T. M., Na, J., Lerman, J. A., Nam, H., Feist, A. M., and Palsson, B. Ø. (2011). A comprehensive genome-scale reconstruction of escherichia coli metabolism—2011. *Molecular systems biology*, 7(1):535.

Orth, J. D., Thiele, I., and Palsson, B. Ø. (2010). What is flux balance analysis? *Nature biotechnology*, 28(3):245–248.

Ostrowski, M., Paulevé, L., Schaub, T., Siegel, A., and Guziolowski, C. (2016). Boolean network identification from perturbation time series data combining dynamics abstraction and logic programming. *Biosystems*, 149:139–153. Selected papers from the Computational Methods in Systems Biology 2015 conference.

Ostrowski, M. and Schaub, T. (2012). ASP modulo CSP: the clingcon system. *TPLP*, 12(4-5):485–503.

Oyarzún, D. A., Chaves, M., and Hoff-Hoffmeyer-Zlotnik, M. (2012). Multistability and oscillations in genetic control of metabolism. *Journal of Theoretical Biology*, 295:139–153.

Passi, A., Tibocha-Bonilla, J. D., Kumar, M., Tec-Campos, D., Zengler, K., and Zuniga, C. (2021). Genome-scale metabolic modeling enables in-depth understanding of big data. *Metabolites*, 12(1):14.

Paulevé, L., Kolčák, J., Chatain, T., and Haar, S. (2020). Reconciling qualitative, abstract, and scalable modeling of biological networks. *Nature communications*, 11(1):4256.

Peyraud, R., Cottret, L., Marmiesse, L., and Genin, S. (2018). Control of primary metabolism by a virulence regulatory network promotes robustness in a plant pathogen. *Nature communications*, 9(1):418.

Prigent, S., Frioux, C., Dittami, S. M., Thiele, S., Larhlimi, A., Collet, G., Gutknecht, F., Got, J., Eveillard, D., Bourdon, J., et al. (2017). Meneco, a topology-based gap-filling tool applicable to degraded genome-wide metabolic networks. *PLoS computational biology*, 13(1):e1005276.

Razzaq, M., Paulevé, L., Siegel, A., Saez-Rodriguez, J., Bourdon, J., and Guziolowski, C. (2018). Computational discovery of dynamic cell line specific boolean networks from multiplex time-course data. *PLOS Computational Biology*, 14(10):e1006538.

Réda, C. and Delahaye-Duriez, A. (2022). Prioritization of candidate genes through boolean networks. In *International Conference on Computational Methods in Systems Biology*, pages 89–121. Springer.

Reynolds, A., King, T., and Kuncak, V. (2017). Solving quantified linear arithmetic by counterexample-guided instantiation. *Formal Methods in System Design*, 51:500–532.

Saez-Rodriguez, J., Alexopoulos, L. G., Epperlein, J., Samaga, R., Lauffenburger, D. A., Klamt, S., and Sorger, P. K. (2009). Discrete logic modelling as a means

to link protein signalling networks with functional analysis of mammalian signal transduction. *Mol Syst Biol*, 5(1):331.

Salgado, H., Gama-Castro, S., Lara, P., Mejia-Almonte, C., Alarcón-Carranza, G., López-Almazo, A. G., Betancourt-Figueroa, F., Peña-Loredo, P., Alquicira-Hernández, S., Ledezma-Tejeida, D., Arizmendi-Zagal, L., Mendez-Hernandez, F., Diaz-Gomez, A. K., Ochoa-Praxedis, E., Muñiz-Rascado, L. J., García-Sotelo, J. S., Flores-Gallegos, F. A., Gómez, L., Bonavides-Martínez, C., del Moral Chávez, V. M., Hernández-Alvarez, A. J., Santos-Zavaleta, A., Capella-Gutierrez, S., Gelpi, J. L., and Collado-Vides, J. (2023). RegulonDB v12.0: a comprehensive resource of transcriptional regulation in E. coli K-12. *Nucleic Acids Research*, 52(D1):D255–D264.

Schaub, T. and Thiele, S. (2009). Metabolic network expansion with answer set programming. In Hill, P. M. and Warren, D. S., editors, *Logic Programming*, pages 312–326, Berlin, Heidelberg. Springer Berlin Heidelberg.

Schuster, S. and Hilgetag, C. (1994). On elementary flux modes in biochemical reaction systems at steady state. *Journal of Biological Systems*, 2(02):165–182.

Shlomi, T., Eisenberg, Y., Sharan, R., and Ruppin, E. (2007). A genome-scale computational study of the interplay between transcriptional regulation and metabolism. *Molecular systems biology*, 3(1):101.

Terfve, C., Cokelaer, T., Henriques, D., MacNamara, A., Goncalves, E., Morris, M. K., Iersel, M. v., Lauffenburger, D. A., and Saez-Rodriguez, J. (2012). Cellnoptr: a flexible toolkit to train protein signaling networks to data using multiple logic formalisms. *BMC systems biology*, 6:1–14.

Thiele, I. and Palsson, B. Ø. (2010). A protocol for generating a high-quality genome-scale metabolic reconstruction. *Nature protocols*, 5(1):93–121.

Thomas, R. (1973). Boolean formalization of genetic control circuits. *Journal of Theoretical Biology*, 42(3):563–585.

Thomas, R. (1981). On the relation between the logical structure of systems and their ability to generate multiple steady states or sustained oscillations. In Della Dora, J., Demongeot, J., and Lacolle, B., editors, *Numerical Methods in the Study of Critical Phenomena*, pages 180–193, Berlin, Heidelberg. Springer Berlin Heidelberg.

Thuillier, K., Baroukh, C., Bockmayr, A., Cottret, L., Paulevé, L., and Siegel, A. (2021). Learning boolean controls in regulated metabolic networks: a case-study. In *Computational Methods in Systems Biology: 19th International Conference,*

CMSB 2021, Bordeaux, France, September 22–24, 2021, Proceedings 19, pages 159–180. Springer.

Thuillier, K., Baroukh, C., Bockmayr, A., Cottret, L., Paulevé, L., and Siegel, A. (2022). MERRIN: MEtabolic regulation rule INference from time series data. *Bioinformatics*, 38(Supplement_2):ii127–ii133.

Thuillier, K., Siegel, A., and Paulevé, L. (2024). Cegar-based approach for solving combinatorial optimization modulo quantified linear arithmetics problems. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(8):8146–8153.

Tournier, L., Goelzer, A., and Fromion, V. (2017). Optimal resource allocation enables mathematical exploration of microbial metabolic configurations. *Journal of mathematical biology*, 75(6):1349–1380.

Trinh, H.-C. and Kwon, Y.-K. (2021). A novel constrained genetic algorithm-based Boolean network inference method from steady-state gene expression data. *Bioinformatics*, 37(Supplement_1):i383–i391.

Tsiantis, N., Balsa-Canto, E., and Banga, J. R. (2018). Optimality and identification of dynamic models in systems biology: an inverse optimal control framework. *Bioinformatics*, 34(14):2433–2440.

Ullah, E., Yosafshahi, M., and Hassoun, S. (2019). Towards scaling elementary flux mode computation. *Briefings in Bioinformatics*, 21(6):1875–1885.

Vaginay, A., Boukhobza, T., and Smaïl-Tabbone, M. (2021). Automatic synthesis of boolean networks from biological knowledge and data. In Dorronsoro, B., Amodeo, L., Pavone, M., and Ruiz, P., editors, *Optimization and Learning*, pages 156–170, Cham. Springer International Publishing.

Varma, A. and Palsson, B. O. (1994). Stoichiometric flux balance models quantitatively predict growth and metabolic by-product secretion in wild-type Escherichia coli W3110. *Appl Environ Microbiol*, 60(10):3724–3731.

Videla, S., Saez-Rodriguez, J., Guziolowski, C., and Siegel, A. (2017). caspo: a toolbox for automated reasoning on the response of logical signaling networks families. *Bioinf*, page btw738.

von Kamp, A. and Klamt, S. (2014). Enumeration of smallest intervention strategies in genome-scale metabolic networks. *PLOS Computational Biology*, 10(1):1–13.

Walpole, J., Papin, J. A., and Peirce, S. M. (2013). Multiscale computational models of complex biological systems. *Annual Review of Biomedical Engineering*, 15(Volume 15, 2013):137–154.

Wang, R.-S., Saadatpour, A., and Albert, R. (2012). Boolean modeling in systems biology: an overview of methodology and applications. *Physical Biology*, 9(5):055001.

Zañudo, J. G. T., Yang, G., and Albert, R. (2017). Structure-based control of complex networks with nonlinear dynamics. *Proc Natl Acad Sci USA*, 114(28):7234–7239.

Zeljić, A., Wintersteiger, C. M., and Rümmer, P. (2017). An approximation framework for solvers and decision procedures. *Journal of automated reasoning*, 58:127–147.

Žiga Pušnik, Mraz, M., Zimic, N., and Moškon, M. (2022). Review and assessment of boolean approaches for inference of gene regulatory networks. *Heliyon*, 8(8):e10222.

Appendices

In this chapter

A Regulated Metabolic Networks

In this appendix, we describe two regulated metabolic networks of *Escherichia coli* for which no standard encodings were available. Part of this thesis has consisted of cleaning, updating, and understanding these networks. Our models are composed of SBML (metabolic networks) files (Hucka et al., 2003) and SBML-qual (Boolean regulatory networks) files (Chaouiya et al., 2013). They are compatible with the rFBA simulation tool *FlexFlux* (Marmiesse et al., 2015). Files associated with each network are available at `https://github.com/kthuillier/regulated-metabolic-models`.

In Section A.1, we describe the model of core-carbon metabolism introduced in Covert et al. (2001) and used in Chapters III to V. In Section A.2, we describe the medium-scale model of the core metabolism of *Escherichia coli* introduced in (Covert and Palsson, 2002) and used in Chapters IV and V.

A.1 Core-carbon Metabolic Model

Comprehensive description of the Boolean network of the core-carbon metabolism model that we reconstruct. The core-carbon metabolism model is introduced in Covert et al. (2001).

A.1.1 Regulated metabolic network

Description. The model description, as provided in the paper, is shown in Tab. 8 (three first columns). At the metabolic level, it contains 20 reactions and 19 metabolites of which 8 are environmental metabolites. The reaction bounds are defined in the paper. From this description, we generate a metabolic network in SBML format. The metabolic network has been manually encoded into SBML, using the open-source Python library *LibSBML* (Bornstein et al., 2008).
At the regulatory level, there are 4 regulatory proteins and 11 non-constant regulatory rules. This model does not consider genes, only regulatory proteins.

Boolean regulatory network. The Boolean network that we reconstructed from the model description is described by the last column of Tab. 8. The rules are functions of a consistent Boolean regulatory state (Def. 2.1 in Chapter II), *i.e.* given a regulated metabolic steady-state (v, w, x) (Def. 1.6 in Chapter I), x is such that:

— for each external metabolite m of concentration w_m: $(w_m > 0) \iff x_m$;

— for each reaction r with a flux value v_r: $(v_r \neq 0) \iff x_r$.

These Boolean rules have been manually encoded into SBML-qual.

Reaction	Name	Regulation	Boolean function
Metabolic reactions			
-1 A $-$ 1 ATP $+$ B	*R1*		$f_{\mathrm{R1}}(x) = 1$
-1 B $+$ 2 ATP $+$ 3 NADH $+$ 1 C	*R2a*	IF NOT(*RPb*)	$f_{\mathrm{R2a}}(x) = x_{\mathrm{RPb}}$
-1 C $-$ 2 ATP $-$ 3 NADH $+$ 1 B	*R2b*		$f_{\mathrm{R2b}}(x) = 1$
-1 B $+$ 1 F	*R3*		$f_{\mathrm{R3}}(x) = 1$
-1 C $+$ 1 G	*R4*		$f_{\mathrm{R4}}(x) = 1$
-1 G $+$ 0.8 C $+$ 2 NADH	*R5a*	IF NOT (*RPO2*)	$f_{\mathrm{R5a}}(x) = \neg x_{\mathrm{RPO2}}$
-1 G $+$ 0.8 C $+$ 2 NADH	*R5b*	IF *RPO2*	$f_{\mathrm{R5b}}(x) = x_{\mathrm{RPO2}}$
-1 C $+$ 2 ATP $+$ 3 D	*R6*		$f_{\mathrm{R6}}(x) = 1$
-1 C $-$ 4 NADH $+$ 3 E	*R7*	IF NOT (*RPb*)	$f_{\mathrm{R7}}(x) = \neg x_{\mathrm{RPb}}$
-1 G $-$ 1 ATP $-$ 2 NADH $+$ 1 H	*R8a*	IF NOT(*RPh*)	$f_{\mathrm{R8a}}(x) = \neg x_{\mathrm{RPh}}$
-1 H $+$ 1 ATP $+$ 2 NADH $+$ 1 G	*R8b*		$f_{\mathrm{R8b}}(x) = 1$
-1 NADH $-$ 1 O2 $+$ 1 ATP	*Rres*	IF NOT(*RPO2*)	$f_{\mathrm{Rres}}(x) = \neg x_{\mathrm{RPO2}}$
Transport processes			
-1 *Carbon1* $+$ 1 A	*Tc1*		$f_{\mathrm{Tc1}}(x) = 1$
-1 *Carbon2* $+$ 1 A	*Tc2*	IF NOT(*RPc1*)	$f_{\mathrm{Tc2}}(x) = \neg x_{\mathrm{RPc1}}$
-1 D_{ext} $+$ 1 D_{ext}	*Td*		$f_{\mathrm{Td}}(x) = 1$
-1 E_{ext} $+$ 1 E_{ext}	*Te*		$f_{\mathrm{Te}}(x) = 1$
-1 F_{ext} $+$ 1 F	*Tf*		$f_{\mathrm{Tf}}(x) = 1$
-1 H_{ext} $+$ 1 H	*Th*		$f_{\mathrm{Th}}(x) = 1$
-1 *Oxygen* $+$ 1 O2	*To2*		$f_{\mathrm{To2}}(x) = 1$
Maintenance and growth			
-1 C $-$ 1 F $-$ 1 H $-$ 10 ATP $+$ 1 *Biomass*	*Growth*		$f_{\mathrm{Growth}}(x) = 1$
Regulatory proteins			
	RPO2	IF NOT(*Oxygen*)	$f_{\mathrm{RPO2}}(x) = \neg x_{\mathrm{Oxygen}}$
	RPc1	IF *Carbon1*	$f_{\mathrm{RPc1}}(x) = x_{\mathrm{Carbon1}}$
	RPh	IF $(v_{\mathrm{Th}} > 0)$	$f_{\mathrm{RPh}}(x) = x_{\mathrm{Th}}$
	RPb	IF $(v_{\mathrm{R2b}} > 0)$	$f_{\mathrm{RPb}}(x) = x_{\mathrm{R2b}}$

Table 8 – Comprehensive description of the Boolean regulatory rules of the core-carbon metabolism introduced in Covert et al. (2001). The first three columns are shown exactly as described in the aforementioned paper. The last column is our reconstruction of the Boolean regulatory rules. v_{Th} and v_{R2b} are the metabolic flux through the reactions *Th* and *R2b*, respectively. Boolean functions take as input a Boolean regulatory state x consistent with a regulated metabolic steady-state (v, w, x) (Def. 2.1 in Chapter II).

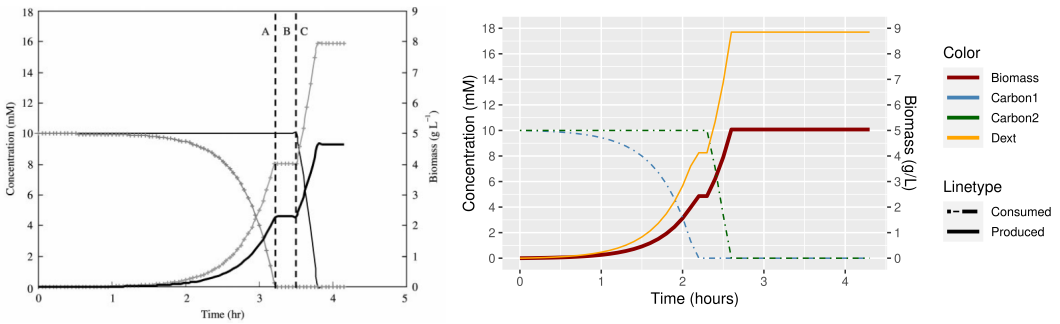
A.1.2 Validation

We validate the reconstructed model using five experimental conditions provided in Covert et al. (2001) and described in Tab. 9.

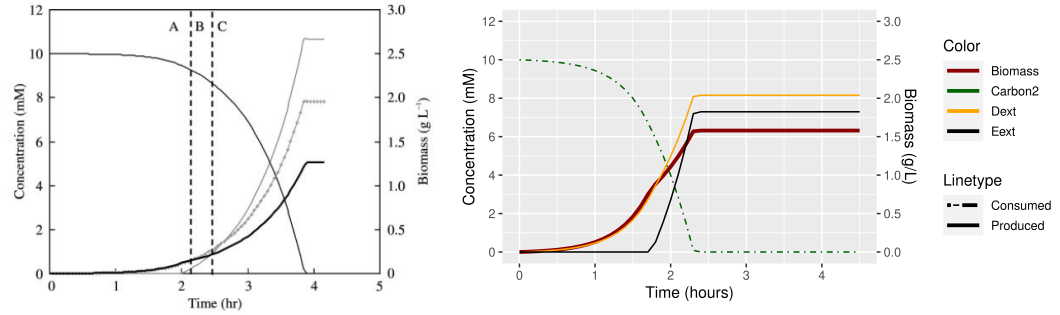
Experiment	Concentration (mM)					Regulatory protein state			
	Carbon1	Carbon2	Oxygen	F_{ext}	H_{ext}	RPO2	RPc1	RPh	RPb
1	10	10	100	0	0	0	1	0	0
2	0	10	5	0	0	0	0	0	0
3	0	10	100	0	2	0	0	1	0
4	0	5	0	0	10	1	0	1	0
5	1	10	100	0.1	5	1	0	1	0

Table 9 – Descriptions of the five experimental conditions used to validate our model of core-carbon metabolism.

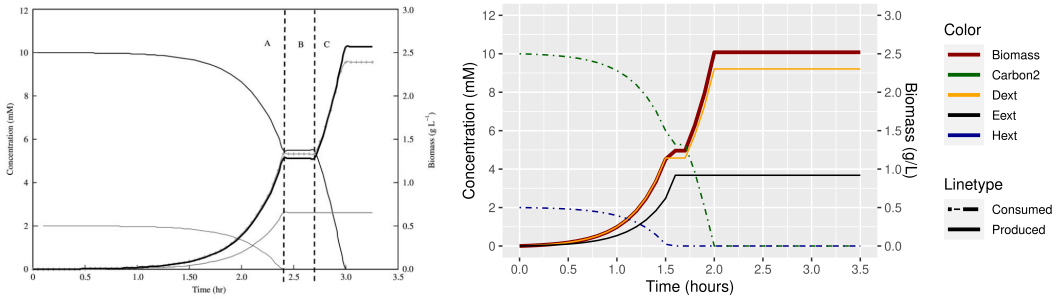
For each experiment, we made an rFBA simulation of the model using FlexFlux Marmiesse et al. (2015) with a timestep of 0.01h, a duration of 3h, and an initial biomass of 0.01g.L^{-1}. A comparison between the rFBA simulations described in Covert et al. (2001) and from our model simulation is shown in Fig. 18. It can be seen that the overall behavior of our model fits with the expected rFBA simulations.



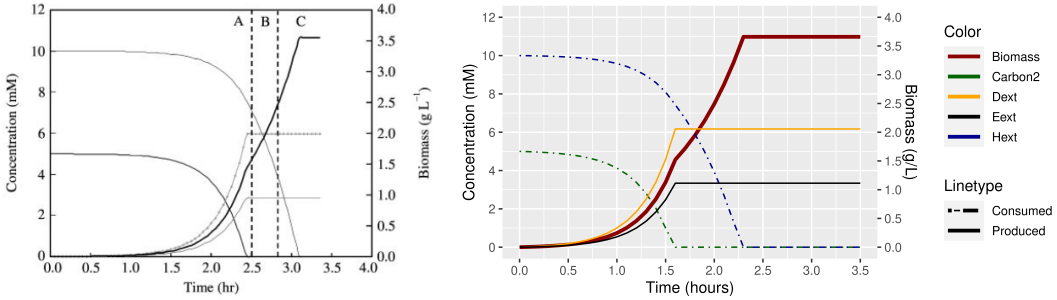
(a) Experiment 1. Left: rFBA simulations from Covert et al. (2001); Right: rFBA simulation made with FlexFlux of our model.



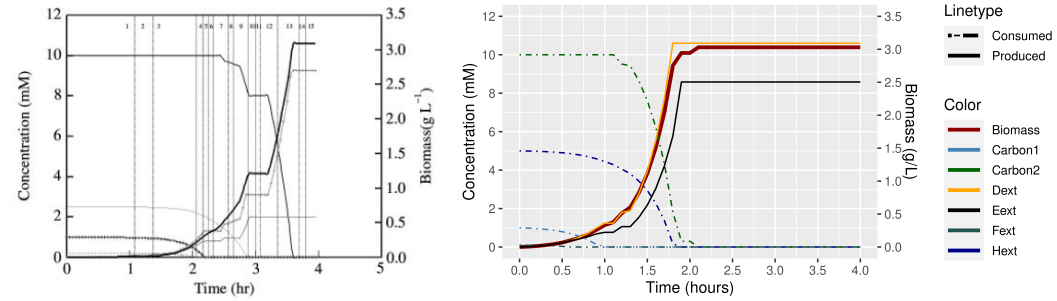
(b) Experiment 2. Left: rFBA simulations from Covert et al. (2001); Right: rFBA simulation made with FlexFlux of our model.



(c) Experiment 3. Left: rFBA simulations from Covert et al. (2001); Right: rFBA simulation made with FlexFlux of our model.



(e) Experiment 4. Left: rFBA simulations from Covert et al. (2001); Right: rFBA simulation made with FlexFlux of our model.



(f) Experiment 5. Left: rFBA simulations from Covert et al. (2001); Right: rFBA simulation made with FlexFlux of our model.

■ **Figure 18** – Side-by-side comparisons of the rFBA simulations provided in Covert et al. (2001) and from our model of the core-carbon metabolism model. For each subfigure, the left graph is from the aforementioned paper, and the right graph is the rFBA simulation made from our reconstructed model. The thick lines (left: black; right: red) are the biomass. Other lines are metabolite concentrations.

Reaction	Protein	Gene	Reaction	Regulatory Logic
ACEA	Isocitrate lyase	*aceA*	ICIT -> GLX + SUCC	IF not (IclR)
ACEB	Malate synthase A	*aceB*	ACCOA + GLX -> COA + MAL	IF not (ArcA or IclR)
ACEE	Pyruvate dehydrogenase	*aceEF, lpdA*	PYR + COA + NAD -> NADH + CO2 + ACCOA	IF (not PdhR)
ACKAR	Acetate kinase A	*ackA*	ACTP + ADP <-> ATP + AC	
ACNAR	Aconitase A	*acnA*	CIT <-> ICIT	IF *(GLCxt or LCTSxt or RIBxt or GLxt or LACxt or PYRxt or SUCCxt or ETHxt or ACxt or FORxt)*
ACNBR	Aconitase B	*acnB*	CIT <-> ICIT	IF *(GLCxt or LCTSxt or RIBxt or GLxt or LACxt or PYRxt or SUCCxt or ETHxt or ACxt or FORxt)*
ACS	Acetyl-CoA synthetase	*acs*	ATP + AC + COA -> AMP + PPI + ACCOA	IF *not (GLCxt or LCTSxt or RIBxt or GLxt or LACxt or PYRxt or SUCxt or ETHxt)* and (not IclR)
ADHER	Acetaldehyde dehydrogenase	*adhE*	ACCOA +2 NADH <-> ETH +2 NAD + COA	IF not (O2xt) or not (O2xt and Cra)
ADK	Adenylate kinase	*adk*	ATP + AMP <-> 2 ADP	
ATPAR	F0F1-ATPase	*atpABCDEFGHI*	ATP <-> ADP + PI +4 HEXT	
CYDA	Cytochrome oxidase bd	*cydAB*	QH2 +.5 O2 -> Q +2 HEXT	IF (not FNR) or ArcA
CYOA	Cytochrome oxidase bo3	*cyoABCD*	QH2 +.5 O2 -> Q +2.5 HEXT	IF not (ArcA or FNR)
DLD1R	D-Lactate dehydrogenase 1	*dld*	PYR + NADH <-> NAD + LAC	
DLD2	D-Lactate dehydrogenase (cyctochrome)	*dld*	LAC + Q -> PYR + QH2	
ENOR	Enolase	*eno*	2PG <-> PEP	
FBAR	Fructose-1,6-bisphosphatate aldolase	*fba*	FDP <-> T3P1 + T3P2	
FBP	Fructose-1,6-bisphosphatase	*fbp*	FDP -> F6P + PI	
FDNG	Formate dehydrogenase-N	*fdnGHI*	FOR + Q -> QH2 + CO2 +2 HEXT	IF FNR
FDOH	Formate dehydrogenase-O	*fdoIHG*	FOR + Q -> QH2 + CO2 +2 HEXT	
FRDA	Fumarate reductase	*frdABCD*	FUM + FADH -> SUCC + FAD	IF FNR or DcuR

Figure 19 – Excerpt of the appendix of Covert and Palsson (2002): First 20 reactions and regulatory rules described in the model of core metabolism.

A.2 Medium-Scale Regulated Metabolic Networks of *Escherichia coli*

Comprehensive description of the Boolean network of the core metabolism of Escherichia coli *that we reconstruct. The model of core metabolism of* Escherichia coli *is introduced in Covert and Palsson (2002).*

A.2.1 Regulated metabolic network

This model is only available in the appendix of Covert and Palsson (2002). An excerpt of the 20 first reactions and regulatory rules of its description is shown in Fig. 19.

At the metabolic level, this model contains 113 reactions over 90 metabolites of which 13 are external metabolites. The metabolic network is shown in Fig. 2 and described in Tab. 10 (last column). We manually encoded the metabolic network into SBML, using the open-source Python library *LibSBML* (Bornstein et al., 2008).

At the regulatory level, there are 86 genes, 20 regulatory proteins, and 203 regulatory rules. A comprehensive description of the Boolean network, as described in the paper, is shown in Tabs. 10 and 11. For the sake of clarity, Boolean rules syntax is simplified: the state x_N of a component N is simply denoted by 'N'; the flux through a reaction R is denoted by '[R $>$ 0]'; and external metabolite availability is denoted by the metabolite name, which ends in 'xt_b'. These Boolean rules have been manually encoded into SBML-qual.

Name	Boolean function	Reaction
Metabolic reactions		
ACEA	aceA	ICIT → GLX + SUCC
ACEB	aceB	ACCOA + GLX → COA + MAL
ACEE	(aceEF ∧ lpdA)	PYR + COA + NAD → NADH + CO2 + ACCOA
ACKAR	ackA	ACTP + ADP ⟺ ATP + AC
ACNAR	acnA	CIT ⟺ ICIT
ACNBR	acnB	CIT ⟺ ICIT
ACS	acs	ATP + AC + COA → AMP + PPI + ACCOA
ADHER	adhE	ACCOA + 2 NADH ⟺ ETH + 2 NAD + COA
ADK	adk	ATP + AMP ⟺ 2 ADP
ATPAR	atpA-I	ATP ⟺ ADP + PI + 4 HEXT
CYDA	cydAB	QH2 + 0.5 O2 → Q + 2 HEXT
CYOA	cyoABCD	QH2 + 0.5 O2 → Q + 2.5 HEXT
DLD1R	dld	NAD + LAC ⟺ PYR + NADH
DLD2	dld	LAC + Q → PYR + QH2
ENOR	eno	_2PG ⟺ PEP
FBAR	fba	FDP ⟺ T3P1 + T3P2
FBP	fbp	FDP → F6P + PI
FDNG	fdnGHI	FOR + Q → QH2 + CO2 + 2 HEXT
FDOH	fdolHG	FOR + Q → QH2 + CO2 + 2 HEXT
FRDA	frdABCD	FUM + FADH → SUCC + FAD
FUMAR	fumA	FUM ⟺ MAL
FUMBR	fumB	FUM ⟺ MAL
FUMCR	fumC	FUM ⟺ MAL
GALER	galE	UDPGAL ⟺ UDPG
GALKR	galK	GLAC + ATP ⟺ GAL1P + ADP
GALM1R	galM	bDGLAC ⟺ GLAC
GALM2R	galM	bDGLC ⟺ GLC
GALTR	galT	GAL1P + UTP ⟺ PPI + UDPGAL
GALUR	galU	G1P + UTP ⟺ UDPG + PPI
GAPAR	gapA	T3P1 + PI + NAD ⟺ NADH + _13PDG
GLK	glk	GLC + ATP → G6P + ADP
GLPA	glpABC	GL3P + Q → T3P2 + QH2
GLPD	glpD	GL3P + Q → T3P2 + QH2
GLPK	glpK	GL + ATP → GL3P + ADP
GLTA	gltA	ACCOA + OA → COA + CIT
GND	gnd	D6PGC + NADP → NADPH + CO2 + RL5P
GPMAR	gpmA	_3PG ⟺ _2PG
GPMBR	gpmB	_3PG ⟺ _2PG
GPSAR	gpsA	GL3P + NADP ⟺ T3P2 + NADPH
ICDAR	icdA	ICIT + NADP ⟺ CO2 + NADPH + AKG
LACZ	lacZ	LCTS → GLC + bDGLAC
MAEB	maeB	MAL + NADP → CO2 + NADPH + PYR
MDHR	mdh	MAL + NAD ⟺ NADH + OA

Name	Boolean function	Reaction
NDH	ndh	NADH + Q → NAD + QH2
NUOA	nuoA-N	NADH + Q → NAD + QH2 + 3.5 HEXT
PCKA	pckA	OA + ATP → PEP + CO2 + ADP
PFKA	pfkA	F6P + ATP → FDP + ADP
PFKB	pfkB	F6P + ATP → FDP + ADP
PFLA	pflAB	PYR + COA → ACCOA + FOR
PFLC	pflCD	PYR + COA → ACCOA + FOR
PGIR	pgi	G6P ⟺ F6P
PGKR	pgk	_13PDG + ADP ⟺ _3PG + ATP
PGL	pgl	D6PGL → D6PGC
PGMR	pgm	G1P ⟺ G6P
PNTA1	pntAB	NADPH + NAD → NADP + NADH
PNTA2	pntAB	NADP + NADH + 2 HEXT → NADPH + NAD
PPA	ppa	PPI → 2 PI
PPC	ppc	PEP + CO2 → OA + PI
PPSA	ppsA	PYR + ATP → PEP + AMP + PI
PTAR	pta	ACCOA + PI ⟺ ACTP + COA
PYKA	pykA	PEP + ADP → PYR + ATP
PYKF	pykF	PEP + ADP → PYR + ATP
RBSK	rbsK	RIB + ATP → R5P + ADP
RPER	rpe	RL5P ⟺ X5P
RPIAR	rpiA	RL5P ⟺ R5P
RPIBR	rpiB	RL5P ⟺ R5P
SDHA1	sdhABCD	SUCC + FAD → FADH + FUM
SDHA2	sdhABCD	FADH + Q ⟺ FAD + QH2
SFCA	sfcA	MAL + NAD → CO2 + NADH + PYR
SUCA	(sucAB ∧ lpdA)	AKG + NAD + COA → CO2 + NADH + SUCCOA
SUCCR	sucCD	SUCCOA + ADP + PI ⟺ ATP + COA + SUCC
TALAR	talA	T3P1 + S7P ⟺ E4P + F6P
TALBR	talB	T3P1 + S7P ⟺ E4P + F6P
TKTA1R	tktA	R5P + X5P ⟺ T3P1 + S7P
TKTA2R	tktA	X5P + E4P ⟺ F6P + T3P1
TKTB1R	tktB	R5P + X5P ⟺ T3P1 + S7P
TKTB2R	tktB	X5P + E4P ⟺ F6P + T3P1
TPIAR	tpiA	T3P1 ⟺ T3P2
ZWFR	zwf	G6P + NADP ⟺ D6PGL + NADPH
ACUPR	1	ACxt + HEXT ⟺ AC
COZTXR	1	CO2xt ⟺ CO2
ETHUPR	1	ETHxt + HEXT ⟺ ETH
FORUPR	focA	FORxt ⟺ FOR
GLCPTS	(ptsGHI ∧ crr)	GLCxt + PEP → G6P + PYR
GLCUP	galP	GLCxt + HEXT → GLC
GLUPR	glpF	GLxt ⟺ GL

Name	Boolean function	Reaction
LACUP	$\neg$ (*GLCxt_ b* $\vee$ *LCTSxt_ b* $\vee$ *RIBxt_ b* $\vee$ *GLxt_ b*)	LACxt + HEXT $\rightarrow$ LAC
LACDN	1	LAC $\rightarrow$ LACxt + HEXT
LACYR	lacY	LCTSxt + HEXT $\iff$ LCTS
O2TXR	1	O2xt $\iff$ O2
PIUP2R	pitAB	PIxt + HEXT $\iff$ PI
PYRUPR	1	PYRxt + HEXT $\iff$ PYR
RIBUPR	rbsABCD	RIBxt + ATP $\rightarrow$ RIB + ADP + PI
DCTAR	dctA	SUCCxt + HEXT $\iff$ SUCC
DCUAR	dcuA	SUCCxt + HEXT $\iff$ SUCC
DCUBR	dcuB	SUCCxt + HEXT $\iff$ SUCC
DCUC	dcuC	SUCC $\rightarrow$ SUCCxt + HEXT
ATPM	1	ATP $\rightarrow$ ADP + PI
Transport processes		
ACex	1	ACxt $\iff$ ACxt_b
CO2ex	1	CO2xt $\iff$ CO2xt_b
ETHex	1	ETHxt $\iff$ ETHxt_b
FORex	1	FORxt $\iff$ FORxt_b
GLCex	1	GLCxt $\iff$ GLCxt_b
GLex	1	GLxt $\iff$ GLxt_b
LACex	1	LACxt $\iff$ LACxt_b
LCTSex	1	LCTSxt $\iff$ LCTSxt_b
O2ex	1	O2xt $\iff$ O2xt_b
PIex	1	PIxt $\iff$ PIxt_b
PYRex	1	PYRxt $\iff$ PYRxt_b
RIBex	1	RIBxt $\iff$ RIBxt_b
SUCCex	1	SUCCxt $\iff$ SUCCxt_b
Maintenance and growth		
Growth	1	Biomass + 13 ATP $\rightarrow$ 13 ADP + 13 PI
VGRO	1	41.25 ATP + 3.54 NAD + 18.22 NADPH + 0.2 G6P + 0.07 F6P + 0.89 R5P + 0.36 E4P + 0.12 T3P1 + 1.49 _3PG + 0.51 PEP + 2.83 PYR + 3.74 ACCOA + 1.78 OA + 1.07 AKG $\rightarrow$ 3.74 COA + 41.25 ADP + 41.25 PI + 3.54 NADH + 18.22 NADP + 1 Biomass

Table 10 – Boolean control rules and reactions of *Escherichia coli* core metabolism (Covert and Palsson, 2002). Element names are used in place of their states, *e.g.* the state of the gene aceA is denoted by 'aceA' instead of x_{aceA}, and activity of a reaction 'R' is denoted by '[R > 0]'. Environmental metabolite names are in italics and end in 'xt_b'.

Name	Boolean function
Regulatory proteins	
ArcA	$\neg$ *O2xt_ b*
Cra	$\neg$ (SurplusFDP $\vee$ SurplusF6P)
Crp	1
DcuR	DcuS
DcuS	*SUCCxt_ b*
FadR	(*GLCxt_ b* $\vee \neg$ *ACxt_ b*)
Fnr	$\neg$ *O2xt_ b*
GalR	$\neg$ *LCTSxt_ b*
GalS	$\neg$ *LCTSxt_ b*
GlpR	$\neg$ *GLxt_ b*
IclR	FadR
Lacl	$\neg$ *LCTSxt_ b*
Mlc	$\neg$ *GLCxt_ b*
PdhR	$\neg$ SurplusPYR
RpiR	$\neg$ *RIBxt_ b*
RbsR	$\neg$ *RIBxt_ b*
SurplusF6P	$\neg$ ([*FBP* > 0] $\wedge \neg$ ([*TKTA2R* > 0] $\vee$ [*TKTB2R* > 0] $\vee$ [*TALAR* > 0] $\vee$ [*TALBR* > 0] $\vee$ [*PGIR* > 0]))
SurplusFDP	$\neg$ ([*FBP* > 0] $\wedge \neg$ ([*TKTA2R* > 0] $\vee$ [*TKTB2R* > 0] $\vee$ [*TALAR* > 0] $\vee$ [*TALBR* > 0] $\vee$ [*PGIR* > 0]))
SurplusPYR	$\neg$ (([*MAEB* > 0] $\vee$ [*SFCA* > 0]) $\wedge \neg$ ([*GLCPTS* > 0] $\vee$ [*PYKF* > 0] $\vee$ [*PYKA* > 0] $\vee$ [*DLD1R* > 0] $\vee$ [*DLD2* > 0] $\vee$ [*DCTAR* > 0] $\vee$ [*DCUAR* > 0] $\vee$ [DCUBR > 0]))
Genes	
aceA	$\neg$ IclR
aceB	$\neg$ (ArcA $\vee$ IclR)
aceEF	$\neg$ PdhR
ackA	1
acnA	(*GLCxt_ b* $\vee$ *LCTSxt_ b* $\vee$ *RIBxt_ b* $\vee$ *GLxt_ b* $\vee$ *LACxt_ b* $\vee$ *PYRxt_ b* $\vee$ *SUCCxt_ b* $\vee$ *ETHxt_ b* $\vee$ *ACxt_ b* $\vee$ *FORxt_ b*)
acnB	(*GLCxt_ b* $\vee$ *LCTSxt_ b* $\vee$ *RIBxt_ b* $\vee$ *GLxt_ b* $\vee$ *LACxt_ b* $\vee$ *PYRxt_ b* $\vee$ *SUCCxt_ b* $\vee$ *ETHxt_ b* $\vee$ *ACxt_ b* $\vee$ *FORxt_ b*)
acs	($\neg$ (*GLCxt_ b* $\vee$ *LCTSxt_ b* $\vee$ *RIBxt_ b* $\vee$ *GLxt_ b* $\vee$ *LACxt_ b* $\vee$ *PYRxt_ b* $\vee$ *SUCCxt_ b* $\vee$ *ETHxt_ b*) $\wedge \neg$ IclR)
adhE	$\neg$ O2xt_b
adk	1
atpA-I	1
crr	1
cydAB	($\neg$ Fnr $\vee$ ArcA)
cyoABCD	$\neg$ (ArcA $\vee$ Fnr)
dctA	($\neg$ (*GLCxt_ b* $\vee$ *LCTSxt_ b* $\vee$ *RIBxt_ b* $\vee$ *GLxt_ b* $\vee$ *LACxt_ b* $\vee$ *PYRxt_ b*) $\wedge \neg$ ArcA $\wedge \neg$ DcuR)
dcuA	1

Name	Boolean function
dcuB	$(\neg\ (GLCxt\_\ b \vee LCTSxt\_\ b \vee RIBxt\_\ b \vee GLxt\_\ b \vee LACxt\_\ b \vee PYRxt\_\ b) \wedge$ Fnr $\wedge$ DcuR)
dcuC	(Fnr $\vee$ ArcA)
dld	1
eno	1
fba	1
fbp	1
fdnGHI	Fnr
fdolHG	1
focA	(ArcA $\vee$ Fnr)
frdABCD	(Fnr $\vee$ DcuR)
fumA	$\neg$ (ArcA $\vee$ Fnr)
fumB	Fnr
fumC	1
galE	$(\neg\ GLCxt\_\ b \wedge \neg$ (GalR $\vee$ GalS))
galK	$(\neg\ GLCxt\_\ b \wedge \neg$ (GalR $\vee$ GalS))
galM	$(\neg\ GLCxt\_\ b \wedge \neg$ (GalR $\vee$ GalS))
galP	$(GLCxt\_\ b \vee LCTSxt\_\ b \vee RIBxt\_\ b \vee GLxt\_\ b \vee LACxt\_\ b \vee PYRxt\_\ b \vee SUCCxt\_\ b \vee ETHxt\_\ b \vee ACxt\_\ b \vee FORxt\_\ b)$
galT	$(\neg\ GLCxt\_\ b \wedge \neg$ (GalR $\vee$ GalS))
galU	1
gapA	1
glk	1
glpABC	$(\neg\ (GLCxt\_\ b \vee LCTSxt\_\ b \vee RIBxt\_\ b) \wedge$ Fnr $\wedge \neg$ GlpR)
glpD	$(\neg\ (GLCxt\_\ b \vee LCTSxt\_\ b \vee RIBxt\_\ b) \wedge \neg$ (ArcA $\vee$ GlpR))
glpF	$(\neg\ (GLCxt\_\ b \vee LCTSxt\_\ b \vee RIBxt\_\ b) \wedge \neg$ GlpR)
glpK	$(\neg\ (GLCxt\_\ b \vee LCTSxt\_\ b \vee RIBxt\_\ b) \wedge \neg$ GlpR)
gltA	1
gnd	1
gpmA	1
gpmB	1
gpsA	1
icdA	1
lacY	1
lacZ	$(\neg\ GLCxt\_\ b \wedge \neg$ Lacl)
lpdA	$\neg$ PdhR
maeB	1
mdh	$\neg$ ArcA
ndh	$\neg$ Fnr
nuoA-N	1
pckA	1
pfkA	1
pfkB	1
pflAB	(ArcA $\vee$ Fnr)
pflCD	(ArcA $\vee$ Fnr)
pgi	1

Name	Boolean function
pgk	$(GLCxt\_b \vee LCTSxt\_b \vee RIBxt\_b \vee GLxt\_b \vee LACxt\_b \vee PYRxt\_b \vee SUCCxt\_b \vee ETHxt\_b \vee ACxt\_b \vee FORxt\_b)$
pgl	1
pgm	1
pitAB	1
pntAB	1
ppa	1
ppc	1
ppsA	Cra
pta	1
ptsGHI	$(((GLCxt\_b \vee LCTSxt\_b \vee RIBxt\_b \vee GLxt\_b \vee LACxt\_b \vee PYRxt\_b \vee SUCCxt\_b \vee ETHxt\_b \vee ACxt\_b \vee FORxt\_b) \wedge \neg \text{Mlc}) \vee ((GLCxt\_b \vee LCTSxt\_b \vee RIBxt\_b \vee GLxt\_b \vee LACxt\_b \vee PYRxt\_b \vee SUCCxt\_b \vee ETHxt\_b \vee ACxt\_b \vee FORxt\_b) \wedge \neg \text{Cra}))$
pykA	1
pykF	$\neg$ Cra
rbsABCD	$(\neg (GLCxt\_b \vee LCTSxt\_b) \wedge \neg \text{RbsR})$
rbsK	$(\neg (GLCxt\_b \vee LCTSxt\_b) \wedge \neg \text{RbsR})$
rpe	1
rpiA	1
rpiB	$\neg$ RpiR
sdhABCD	$\neg$ (ArcA $\vee$ Fnr)
sfcA	1
sucAB	$\neg$ PdhR
sucCD	1
talA	1
talB	1
tktA	1
tktB	1
tpiA	1
zwf	1

Table 11 – Boolean regulatory rules, including feedback rules, of *Escherichia coli* core metabolism (Covert and Palsson, 2002). Element names are used in place of their states, *e.g.* the state of the gene aceA is denoted by 'aceA' instead of x_{aceA}, and activity of a reaction 'R' is denoted by '[R > 0]'. Environmental metabolite names are in italics and end in 'xt_b'.

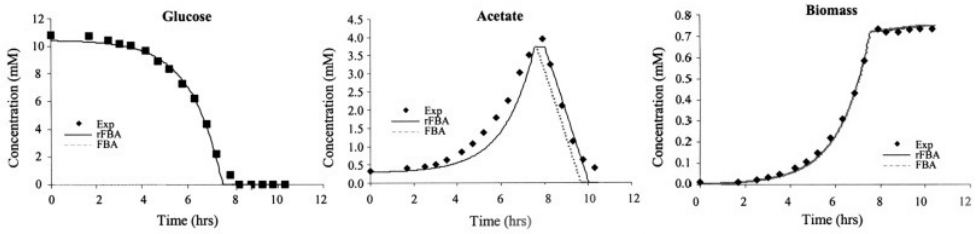
A.2.2 Validation

We validate the reconstructed model using the experimental conditions provided in Covert and Palsson (2002). The three experimental conditions provided in the paper are described in the following table (Tab. 12)

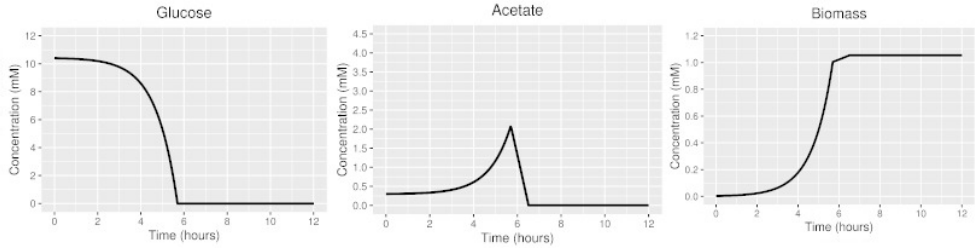
Experiment	Concentration (mM)					Biomass ($g.L^{-1}$)	Duration (h)
	Acetate	Glucose	Lactose	Oxygen	Phosphate		
1	0.3	10.4	0	9999	9999	0.003	12
2	0	10.5	0	9999	9999	0.002	11
3	0	1.6	5.8	9999	9999	0.011	9

Table 12 – Descriptions of the three experimental conditions used to validate our model of *Escherichia coli* core metabolism.

For each experiment, we made an rFBA simulation of our model using FlexFlux (Marmiesse et al., 2015) with a timestep of 0.01h. Figures 20,21, and 22 compare for the three experiments the rFBA simulations described in the aforementioned paper with the rFBA simulations made with our model. The overall dynamics of our model and the one described in the paper are similar.

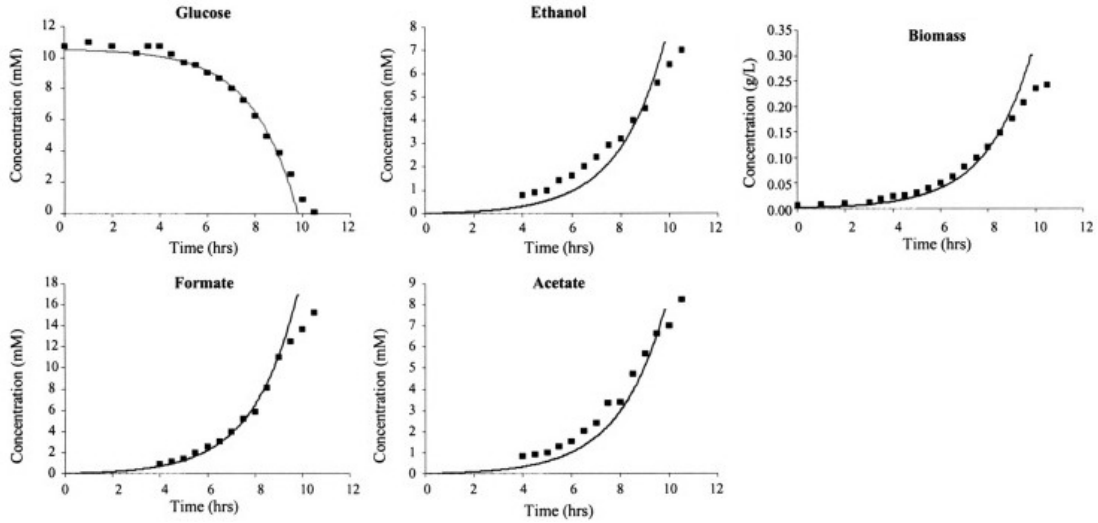


(a) Figures from Covert and Palsson (2002).

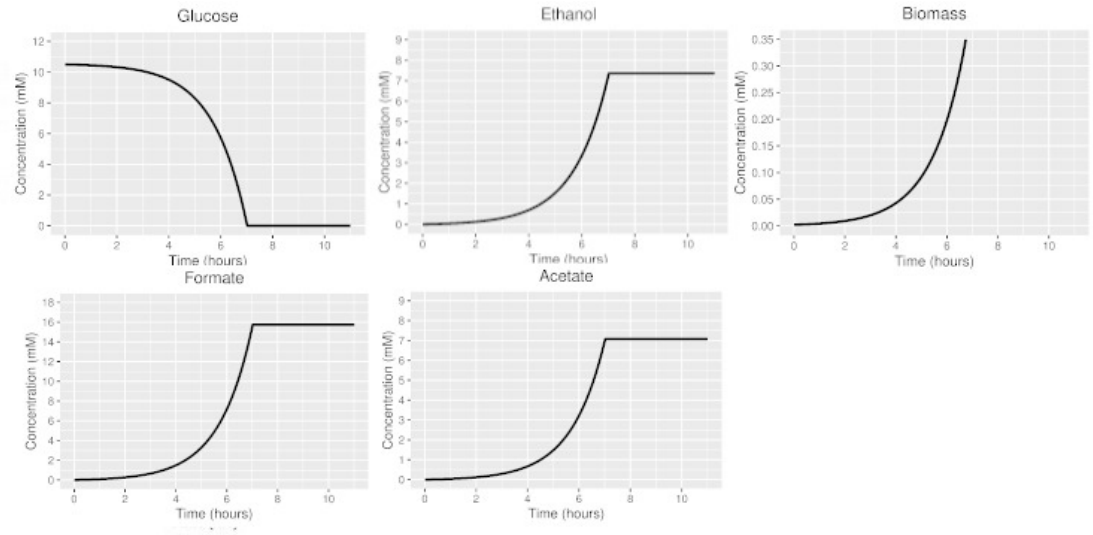


(b) rFBA simulation of our model made with FlexFlux.

Figure 20 – Experiment 1: rFBA simulation of aerobic growth on acetate with glucose reutilization of the medium-scale model. The graphs show, from left to right, the kinetics of glucose, the kinetics of acetate, and the growth (biomass production).

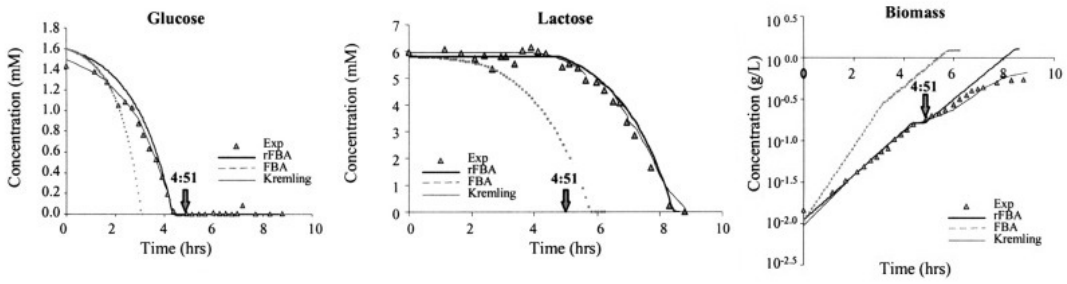


(a) Figures from Covert and Palsson (2002).

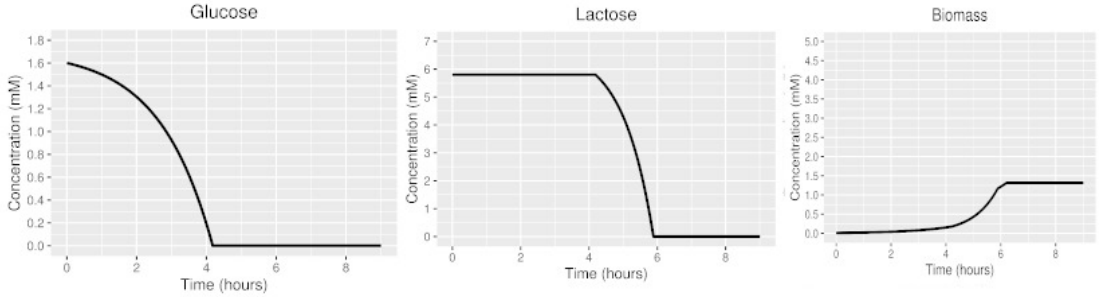


(b) rFBA simulation of our model made with FlexFlux.

■ **Figure 21** – Experiment 2: rFBA simulation of anaerobic growth on glucose of the medium-scale model. The graphs show, from left to right, the kinetics of glucose, the kinetics of ethanol, the growth (biomass production), the kinetics of formate, and the kinetics of acetate.



(a) Figures from Covert and Palsson (2002).



(b) rFBA simulation of our model made with FlexFlux.

■ **Figure 22** – Experiment 3: rFBA simulation of aerobic growth on glucose and lactose of the medium-scale model. The graphs show, from left to right, the kinetics of glucose, the kinetics of lactose, and the growth (biomass production).

B ASP Programs for Addressing the Inference Problems

In this appendix, we present the ASP program used to solve the different formulations of the inference problem. Section B.1 gives the ASP representation of the inference problem inputs. Then in Section B.2 and Section B.3, we describe the ASP programs used to solve the relaxed inference problem and hybrid inference problem formulation, respectively.

B.1 Encoding of the Inference Problem Inputs

B.1.1 Metabolic network

Given a metabolic network $\mathcal{N} = (\mathcal{M}_{\text{ext}})$, the metabolic network $\mathcal{N}$ is represented such that each reaction $r \in \mathcal{R}$ is modeled by a set of facts of the form:

`reactant(m,r,s)` for each reactant metabolite $m \in \mathcal{M}$ of r, that is, $s = S_{mr} < 0$.

`product(m,r,s)` for each product metabolite $m \in \mathcal{M}$ of r, that is, $s = S_{mr} > 0$.

`ext(m)` encode external metabolites $m \in \mathcal{M}_{\text{ext}}$.

`obj(r)` encode the growth reaction $r \in \mathcal{R}$. There is exactly one growth reaction in the ASP program.

`bounds(r,lr,ur)` encode the thermodynamics bounds l_r, u_r of each reaction $r \in \mathcal{R}$.

`rev(rf,rr)` encode the reversible reactions r, that is, reactions with $l_r < 0 < u_r$. In practice, the reversible reactions are split into two reactions: a forward reaction (`rf`) and a reversible reaction (`rr`) in our ASP encoding. Their bounds are such that: $(l_{rf}, u_{rf}) = (0, u_r)$ and $(l_{rr}, u_{rr}) = (0, -l_r)$.

In practice, predicates `ext/1`, `obj/1`, `bounds/3`, and `rev/2` are not used for the relaxed inference problem.

Example. The objective reaction `Growth` and external metabolites of the *toy* model metabolic network (Fig 3) is described by the following set of facts.

```
1 reactant("A","Growth","1"). reactant("ATP","Growth","1").
2 product("Biomass","Growth","1"). product("NADH","Growth","1").
3
4 bounds("Growth","0","9999"). obj("Growth").
5
6 ext("Carbon1"). ext("Carbon2"). ext("Oxygen").
7 ext("Biomass"). ext("Dext"). ext("Eext").
```

B.1.2 Prior Knowledge Network

The prior knowledge network $\mathcal{G} = (V, E)$ is defined using three predicates: `node/1`, `in/3` and `maxC/2`. It is modeled such that:

`node/1` denotes nodes in V, that is, $\forall n \in V$ there is a fact `node(n)`.

`in/3` denotes signed interactions between nodes, that is, $\forall (u, s, v) \in E$ there is a fact `in(u, v, s)`.

`maxC/2` denotes the maximum number of clauses in *disjunctive normal form* that can have a rule. As we are looking for monotone BNs, we can pre-compute the maximum number of conjunctive clauses in rules. The fact `maxC(v,c)` maps a node $v \in V$ to its maximum number of conjunctive clauses c. In practice, c is defined as $\lceil \binom{n}{n/2} \rceil$.

Example. The prior knowledge network of the *toy* model (Figure 3b in Thuillier et al. (2021) - Chapter III) is defined by:

```
1  node("RPcl"). node("RPO2").
2  {node("Carbon1")}. {node("Carbon2")}. {node("Oxygen")}.
3  {node("Rres")}. {node("Tc2")}. {node("Tc1")}.
4
5  in("Carbon1","RPcl",(-1;1)). in("RPcl","Tc2",(-1;1)).
6  in("RPcl","Tc1",(-1;1)). in("Carbon2","RPcl",(-1;1)).
7  in("Oxygen","RPO2",(-1;1)). in("RPO2","Rres",(-1;1)).
8  in("Rres","RPO2",(-1;1)). in("Tc2","RPcl",(-1;1)).
9  in("Tc1","RPcl",(-1;1)).
10
11 maxC("Carbon1",1). maxC("Carbon2",1). maxC("Oxygen",1).
12 maxC("RPcl",6). maxC("RPO2",2). maxC("Rres",1).
13 maxC("Tc2",1). maxC("Tc1",1).
```

B.1.3 Time series observations

The sequence of observations T is a set of pairs (s^{t1}, s^{t2}) where s^{t1} and s^{t2} are regulated metabolic steady-states. Two predicates are used to encode these observations: `next/2` and `obs/3`. Facts of the form:

`next(t1,t2)` represent that time $t1$ and $t2$ are successive (but not necessarily consecutive) timesteps.

`obs(t,v,s)` model that at timestep t, the component v is observed in state s, with $s = 1$ is the v is active/available and $s = -1$ if v is inactive/unavailable.

`obj(t,s)` model that at timestep t, the observed growth is s.

Example. Consider the two first timesteps ($t1 = (1, 0)$ and $t2 = (1, 1)$) of experiment 1 described in Thuillier et al. (2021) - Chapter III:

	External metabolites			Regulatory proteins		Reactions								
Time	w_{Carbon1}	w_{Carbon2}	w_{Oxygen}	x_{RPO2}	x_{RPcl}	v_{Tc1}	v_{Tc2}	v_{To2}	v_{Td}	v_{Te}	v_{Growth}	v_{Rres}	v_{R6}	v_{R7}
(1, 0)	20	20	100	0	1	0	0	0	0	0	0	0	0	0
(1, 1)	20	20	100	0	1	10.5	0	10.5	0	0	10.5	10.5	0	0

Their ASP representation is:

```
1  obs((1,0),"Carbon1",1). obs((1,0),"Carbon2",1).
2  obs((1,0),"Growth",-1). obs((1,0),"Oxygen",1).
3  obs((1,0),"R6",-1). obs((1,0),"R7",-1). obs((1,0),"RPO2",-1).
4  obs((1,0),"RPcl",1). obs((1,0),"Rres",-1). obs((1,0),"Tc1",-1).
5  obs((1,0),"Tc2",-1). obs((1,0),"Td",-1). obs((1,0),"Te",-1).
6  obs((1,0),"To2",-1).
7  obj((1,0),"0").
8
9  obs((1,1),"Carbon1",1). obs((1,1),"Carbon2",1).
10 obs((1,1),"Growth",1).obs((1,1),"Oxygen",1).
11 obs((1,1),"R6",-1). obs((1,1),"R7",-1). obs((1,1),"RPO2",-1).
12 obs((1,1),"RPcl",1). obs((1,1),"Rres",1). obs((1,1),"Tc1",1).
13 obs((1,1),"Tc2",-1). obs((1,1),"Td",-1). obs((1,1),"Te",-1).
14 obs((1,1),"To2",1).
15 obj((1,1),"10.5").
16
17 next((1,0),(1,1)).
```

Note that special reaction bounds can be provided to the program using:

`bounds(t,r,lr,ur)` models that at time t, the reaction r should have the bounds lr, ur. Facts of this form are used to represent external metabolite availability by adding special bounds to exchange reactions.

`param(transport,exp,r,lr,ur)` models that for all observations of the experiment exp, the reaction r should use the bounds lr, ur instead of the metabolic networks bounds.

In practice, the predicates `obj/2`, `bounds/4`, and `param/5` are not used in the encoding of the relaxed inference problem.

B.1.4 Boolean objective function

Used for the relaxed inference problem (Chapter III) only.

The Boolean objective function $\hat{o}$ is encoded with the predicate `score/3`, that associate a timestep to a score $\hat{o}(x)$ where x is the Boolean metabolic steady-state. Facts derived from this predicate have the form `score(t,a,s)` where t is a timestep,

$s = \hat{o}(x)$ is an integer, and $a \in \{o, v\}$ is the score source: the observed metabolic steady-state ($a = o$) or the inferred metabolic steady-states ($a = v$). Note that no facts are initially used to define the score function.

Example. In Thuillier et al. (2021) - Chapter III, we consider the Boolean objective function $\hat{o}(x) = x_{\text{Tc1}} + x_{Tc2} + x_{To2}$. In ASP, it is modeled by:

```
1 opt(T) :- T=("Tc1";"Tc2";"To2").
2 score_z(T,N,0) :- z(T,N,-1). score_z(T,N,1) :- z(T,N,1).
3 score(T,o,S) :- time(T), S=#sum{V,N: opt(N), score_z(T,N,V)}.
4 score(T,v,S) :- time(T), S=TC1+TC2+TO2, score_z(T,"Tc1",TC1),
5                 score_z(T,"Tc2",TC2), score_z(T,"To2",TO2).
```

B.2 Relaxed Inference Problem

In this section, we present the ASP encoding used to address the relaxed inference problem. It relies on the saturation methods. The ASP encoding is shown below.

```
1  {clause(N,1..C,L,S): in(L,N,S), maxC(N,C), node(N)}.
2
3  :- clause(N,_,L,S), clause(N,_,L,-S).
4  1 {constant(N,(-1;1))} 1 :- node(N), not clause(N,_,_,_).
5  constant(N) :- constant(N,_).
6
7  size(N,C,X) :- X = #count{L, S: clause(N,C,L,S)}, clause(N,C,_,_).
8  :- clause(N,C,_,_), not clause(N,C-1,_,_), C > 1.
9  :- size(N,C1,X1), size(N,C2,X2), X1 < X2, C1 > C2.
10 clausediff(N,C1,C2,L) :- clause(N,C1,L,_), not clause(N,C2,L,_),
11                          clause(N,C2,_,_), C1 != C2.
12 mindiff(N,C1,C2,L) :- clausediff(N,C1,C2,L),
13                       L <= L': clausediff(N,C1,C2,L'),
14                                clause(N,C1,L',_), C1!=C2.
15 :- size(N,C1,X), size(N,C2,X), C1 > C2,
16    mindiff(N,C1,C2,L1), mindiff(N,C2,C1,L2), L1 < L2.
17 :- size(N,C1,X1), size(N,C2,X2), C1 != C2, X1 <= X2,
18    clause(N,C2,L,S): clause(N,C1,L,S).
```

Encoding DNF formulas. Lines 1–18 define rules encoding logical formulas in disjunctive normal form (DNF) as proposed in *BoNesis* (Chevalier et al., 2020). A conjunctive clause of a DNF formula is defined using the predicate `clause/4`. The set of admissible clauses according to the input prior knowledge network delimiting the solution space is defined in Line 1. As we focus on inferring locally monotone BNs, we want a node n to be of a single sign (n or $\neg n$) in local functions (Line 3). A node is set to a constant value, either 1 or -1, if its local function does not

contain any clauses (Lines 4–5). Thus, each node n matches either with a constant atom `constant(n)` or a set of clause atoms of the form `clause(n, id, v, s)`, where id is the clause ID, v is another node and $s \in \{-1; 1\}$ is the sign of v in the clause id of local function of the node n.

With current BN representation, there exist many semantically identical Boolean formulas, *e.g.* `clause("A", 1, "B", 1)` is semantically equivalent to `clause("A", 2, "B", 1)`. Lines 7-18 allow removing these duplicates, thus reducing the solution space. These lines define an order on clauses, based on 3 criterion:

1. if a clause c_n is non-empty, then all the clauses $\{c_1, ..., c_{n-1}\}$ must be non-empty (Line 8);

2. if a clause c_i contain n_i elements and a clause c_j contain n_j elements, with $n_i < n_j$, then $i < j$ (Lines 9-16);

3. a clause c_i of a local function f_i could not strictly include another clause c_j (with $i \neq j$) of f_i: $c_i \not\subseteq c_j$ (Lines 17-18).

Learning Boolean networks. Notice that at this point, we have a representation of the search space $\mathbb{F}(\mathcal{G})$ of BNs delimiting by the prior knowledge network $\mathcal{G}$.

```
19 update(T1,A) :- mode(T1,reg), node(A), not inp(A,_).
20 mode(T1,reg) :- next(T1,_).
21
22 constant(A,-1) :- inp(A,_).
23 :- constant(A), not inp(A,_).
24
25 eval(T,A,C,-1) :- update(T,A), clause(A,C,L,V), read(T,L,-V).
26 eval(T,A,C,1) :- read(T,L,V): clause(A,C,L,V);
27                   update(T,A), clause(A,C,_,_).
28 eval(T,A,1) :- eval(T,A,C,1), clause(A,C,_,_).
29 eval(T,A,-1) :- eval(T,A,C,-1): clause(A,C,_,_);
30                 update(T,A), clause(A,C,_,_).
31 eval(T,A,V) :- update(T,A), constant(A,V).
32
33 x(T2,A,V) :- inp(A,_), next(_,T2), v(T2,A,V).
34 x(T2,A,V) :- next(T1,T2), not inp(A,_), not update(T1,A), v(T1,A,V).
35 x(T2,A,V) :- next(T1,T2), update(T1,A), eval(T1,A,V).
```

Lines 20–32 define the Boolean network dynamics. Line 20 defines the update mode of the BN as synchronous, *i.e.* all the local functions of the BN are applied at each timestep. The predicate `update/2` matches the set of local functions that must be applied to each timestep. As we rely on a synchronous update, for each timestep t and each node n which is not an input, the model has the atom `update(t,n)`. Notice that local functions associated with input metabolites are never applied. Input metabolites are associated with constant functions, defined in Lines 23–24,

as the input metabolites depend on external observations independent of the rest of the system.

Lines 26-32 encode valid transitions between regulatory states. From a given regulatory state x modeled with the predicate `read/3`, it applies one synchronous update. Given f a BN, the new regulatory state is represented with the predicate `eval/3`, namely `eval(t,n,v)` for a timestep t the node n has a Boolean value $f_n(x) = v$. As some node states could be fixed (force-activated or force-inhibited depending on the input observations), it is necessary to only update the value of free nodes. The values of non-updated nodes, as input metabolites, are copied from the initial Boolean state x (Line 34-35). At time t the updated value v of a free node n is copied into atoms of the form `x(t,n,v)` (Line 36).

Modeling Boolean rFBA dynamics. The initial Boolean states x, used to compute the transitions from, are defined with `read/3`, namely `read(t,n,v)` with t a timestep, n a node and $v \in \{-1; 1\}$ an initial state. Consider two regulated Boolean metabolic steady-states s and s', s' succeeds to s if and only if $\exists\, x = (s'_{\mathcal{M}_{\text{ext}}}, s_{\mathcal{R}\cup\mathcal{P}})$ such that $s' \in \text{next}^{\mathbb{B}}_{(\mathcal{N},\mathcal{P},f,\hat{\phi})}(x)$ (Chapter III).

```
38 read(T,A,V) :- next(T,_), not inp(A,_), v(T,A,V).
39 read(T,A,V) :- next(T,T2), inp(A,_), v(T2,A,V).
```

Lines 35–36 initialize the initial Boolean state x according to this definition. In this case, if for each timestep the synchronous update of x leads to s', then the current Boolean network is admissible. It allows explaining the input observations given the Boolean rFBA semantics.

Encoding Boolean metabolic steady-states. At this point, we have a representation of the search space $\mathbb{F}(\mathcal{G})$ and can simulate synchronous Boolean networks. To model the Boolean satisfiability problem, one must encode Boolean metabolic steady-states into the ASP model.

```
41 inp(X,R) :- reactant(X,R), not product(X,_).
42 r(r,A,R) :- reactant(A,R), product(A,_). r(p,A,R) :- product(A,R),
43                                                      reactant(A,_).
44 varm(A) :- r(_,A,_). varm(A) :- r(_,_,A). varm(A) :- inp(A,_).
45 time(T1) :- next(T1,_). time(T2) :- next(_,T2).
46
47 1 { v(T,A,(1;-1)) } 1 :- time(T), varm(A).
48 :- obs(T,A,V), v(T,A,-V).
49 :- time(T), r(S,A,_), v(T,A,1), v(T,R,-1): r(S,A,R).
50 :- time(T), r(_,A,R), v(T,R,1), v(T,A,-1).
51 :- time(T), inp(X,R), v(T,X,-1), v(T,R,1).
52
53 varx(A) :- node(A), not varm(A).
54 1{v(T,A,(-1;1))}1 :- varx(A), time(T).
```

```
55 :- varx(A), x(T,A,V), v(T,A,-V).
56 :- x(T,A,-1), v(T,A,1), node(A).
```

Boolean metabolic steady-states are encoded using the logical constraints Lines 41–51. This encoding relies on the predicate `v/3`, namely `v(t,n,v)` matching a Boolean state v to a component (metabolite or reaction) n at a given timestep t (Line 47). The observed components are fixed to their observed states (Line 48), *i.e.* if at time t the reaction Tc1 is not activated then there is `v(t,"Tc1",-1)`. The values of all the non-observed components are set according to the definition of Boolean metabolic steady-states:

1. Line 49: a metabolite is produced if it is consumed by at least one reaction;
2. Lines 50–51: a reaction is activated if all its reactants and products are activated.

Actually, for each timestep t, the set of atoms `v(t,n,v)` must be a Boolean metabolic steady-state.

Lines 53–56 restrict the set of Boolean metabolic steady-states to regulated Boolean metabolic steady-states, that is, for a reaction r to be activated at a timestep t, the node associated with r in the BN must be activated at t too.

Application of the Boolean objective function. It remains to universally quantified logical constraints of the relaxed inference problem:

$$\forall z \in \mathrm{MSS}^{\mathbb{B}}(\mathcal{N}),\ z_{\mathrm{Inp}} \neq x'_{\mathrm{Inp}} \vee z_{\mathcal{P}} \neq f_{\mathcal{P}}(x') \vee \hat{o}(z) \leq \hat{o}(y) \vee (\forall r \in \mathcal{R},\ z_r \not\leq f_r(x'))$$

The observed regulated Boolean metabolic steady-states must be optimal according to the Boolean objective function $\hat{o}$. In other words, for a given regulatory state x, such that the observed state v is a regulated Boolean metabolic steady-state, there is no timestep t where $\hat{o}(v)$ is not optimal. To model this constraint, we seek to solve the inverse problem: to find a regulated Boolean metabolic steady-state such that $\hat{o}(z) > \hat{o}(v)$ and to prohibit the existence of such a z.

```
58 z(T,A,1);z(T,A,-1) :- time(T), varm(A).
59
60 no_rmss(T) :- inp(A,_), v(T,A,V), z(T,A,-V).
61 no_rmss(T) :- time(T), r(S,A,_), z(T,A,1), z(T,R,-1): r(S,A,R).
62 no_rmss(T) :- time(T), r(_,A,R), z(T,R,1), z(T,A,-1).
63 no_rmss(T) :- time(T), inp(X,R), z(T,X,-1), z(T,R,1).
64
65 no_rmss(T):- varx(A), x(T,A,V), z(T,A,-V).
66 no_rmss(T) :- x(T,A,-1), z(T,A,1), node(A).
67
```

```
68 valid(T) :- time(T), no_rmss(T).
69 valid(T) :- time(T), score(T,v,V), score(T,o,O), V <= O.
70
71 z(T,A,-V) :- time(T), varm(A), z(T,A,V), valid(T).
72 :- next(_, T), time(T), not valid(T).
```

Lines 58–66 define regulated Boolean metabolic steady-states z as previously. There are two main differences:

— The observed values are not considered. We want to see if there are regulated Boolean metabolic steady-states that are better, according to the objective function than the observed one.

— The predicate `no_rmss/1` allows noting that a Boolean state is not a regulated Boolean metabolic steady-state for a given timestep t: it either does not match the input metabolites or is not a regulated Boolean metabolic steady-state.

As we are looking for a regulated Boolean metabolic steady-state z such that $\hat{o}(z) > \hat{o}(v)$ with v the observed regulated Boolean metabolic steady-state, we need to define *valid* Boolean states:

— Line 68 defines as valid all the Boolean states that do not satisfy the definition of regulated Boolean metabolic steady-states. This may seem counter-intuitive, but such Boolean states are not candidates for z. Declaring them *valid* means that you don't have to worry about their score, they are simply ignored.

— Line 65 introduces the Boolean objective function $\hat{o}$ encoded with the predicate `score/3`. A regulated Boolean metabolic steady-state is valid if it has a score less than or equal to the score of the observed state v.

All the non-valid regulated Boolean metabolic steady-state z are computed through the saturation technique (Eiter et al., 2009; Gebser et al., 2011) (Chapter III Section 2.2). The disjunctive variables z are defined in Line 58. The existence of non-valid states is prohibited by Lines 71–72. In particular, Line 71 saturates the set of all valid states to ensure that non-valid ones are computed and returned if they exist. Line 72 ensures that no not-saturated state exists, *i.e.* that all states are *valid*.

Display answer sets. Finally, Lines 74–75 allows displaying only the `clause/4` predicates when showing answer sets.

```
74 #show.
75 #show clause/4.
```

B.3 Hybrid Inference Problem

In this section, we present the MerrinASP (Thuillier et al., 2024) encoding used to address the hybrid inference problem (Chapter V).

First, in Section B.3.1, we present the extended ASP syntax introduced by *MerrinASP* to model quantified linear constraints. Then, in Section B.3.2, we present the ASP program modeling the combinatorial over-approximation of the hybrid inference problem. This is also the ASP program used in *MERRIN*. Finally, in Section B.3.3, we present the quantified linear constraints modeled in *MerrinASP* syntax.

B.3.1 MerrinASP Extended Syntax

MerrinASP extends the ASP syntax to model linear constraints with one level of quantifier. In particular, it introduces five new "commands", known as *theory atoms*:

1. `&dom[id]{lb..ub} = v`: define the range of the real-valued variable $v \in \mathbb{R}$ such that $lb \leq v \leq ub$, with $lb, ub \in \mathbb{R}$.

2. `&sum[id]{k1 * v1; ...; kn * vn} ◇ b`: define a linear constraint of the form $\sum_{i=1}^{n} k_i \times v_i \diamond b$, where $k_i \in \mathbb{R}$ are coefficients, $v_i \in \mathbb{R}$ are variables, $\diamond \in \{\leq, \geq, =\}$ and $b \in \mathbb{R}$ is the constraint's bound.

3. `&maximize[id]{k1 * v1; ...; kn * vn}`: define the objective function of the linear optimization problem as the maximization of $\sum_{i=1}^{n} k_i \times v_i$, where $k_i \in \mathbb{R}$ are coefficients and $v_i \in \mathbb{R}$ are variables. This theory atom is used only to compute the assignments of the real-valued variables for display after an answer set is computed.

4. `&minimize[id]{k1 * v1; ...; kn * vn}`: similar to `&maximize`, but it defines the objective function as the minimization of $\sum_{i=1}^{n} k_i \times v_i$.

5. `&assert[id]{k1 * v1; ...; kn * vn} ◇ b`: define a universal linear constraint of the form $\sum_{i=1}^{n} k_i \times v_i \diamond b$, where $k_i \in \mathbb{R}$ are coefficients, $v_i \in \mathbb{R}$ are variables, $\diamond \in \{\leq, \geq, =\}$ and $b \in \mathbb{R}$ is the constraint's bound. This constraint is satisfied if and only if all real-valued assignments that satisfy all constraints `&dom[id]` and `&sum[id]` satisfied $\sum_{i=1}^{n} k_i \times v_i \diamond b$.

For each theory atom, the argument `id` is optional, if not provided it is set to a default value `default`. The `id` argument allows for partitioning the set of linear constraints. When ensuring the satisfiability of a set of linear constraints, we only ensure that the set of linear constraints in the same partition are satisfiable (*i.e.* linear constraints with the same `id` value).

Comparison with clingo[LP] extended syntax. The extended ASP syntax of *MerrinASP* is quite similar to the one used for *clingo[LP]* (Janhunen et al., 2017) and *clingo-lpx*[10]. The main difference is that they do not have the `&assert` theory atoms, since they do not support quantified linear constraints. Moreover, they do allow for linear constraints partitioning, therefore their theory atoms do not have the `id` arguments. All extended ASP programs compatible with *clingo[LP]* or *clingo-lpx* are compatible with *MerrinASP*.

B.3.2 Combinatorial Over-Approximation

In this section, we present the ASP program used to address the combinatorial part of the inference problem. It is used in *MERRIN* (Chapter IV), and in the *MerrinASP* implementation of the hybrid inference problem (Chapter V).

Note that this ASP program is quite similar to the ASP program used to encode the relaxed inference problem (Appendix B.2). The main difference is that the hybrid inference problem does not need to infer Boolean networks (BNs) that are exactly compatible with the time series data. Moreover, we introduce more criteria to *break the symmetry* and reduce the number of semantically equivalent BNs that can be inferred.

Non-exact trace compatibility. Unlike for the relaxed inference problem, a maximum size difference between traces and time series $K_{\max}$ should be provided as input. This input is encoded as the fact: `maxGap(Kmax).`

```
1  % Number of timestep per experiment
2  nbObs(E,S) :- obs((E,_),_,_), S=#count{I: obs(T,_,_), T=(E,I)}.
3
4  % Time definition
5  time((E,1..S)) :- obs((E,_),_,_), nbObs(E,S).
6
7  % Added time
8  1 { gap(E,0..K) } 1 :- obs((E,_),_,_), maxGap(K).
9  time((E,S+A)) :- obs((E,_),_,_), nbObs(E,S), gap(E,K), A=0..K.
10
11 % Simulation time must be successive
12 :- time((E,ID)), ID > 1, not time((E,ID-1)).
13
14 % Successive simulation time
15 maxTs(E,S+K) :- nbObs(E,S), gap(E,K).
16
17 % Successive time definition
18 succ((E,T1),(E,T2)) :- maxTs(E,ID), T1=(1..(ID-1)), T2=T1+1,
19                         time((E,T1)), time((E,T2)).
```

[10] Available on https://github.com/potassco/clingo-lpx.

```
20
21 % Mapping between observations and simulation times
22 1 { map(To,Ts): time(Ts), Ts=(E,_) } 1 :- obs(To,_,_), To=(E,ID).
23
24 % Order must be preserved
25 :- map(To,Ts), map(To',Ts'), To=(E,ID), To'=(E,ID'),
26      next(To,To'), Ts > Ts'.
27
28 % Bijective mapping
29 :- map(To,Ts), map(To',Ts'), To = To', Ts != Ts'.
30 :- map(To,Ts), map(To',Ts), To != To'.
31
32 % The first observation is the first simulation time
33 :- next(To,_), not next(_,To), Ts=(E,_), map(To,Ts), Ts!=(E,1).
```

Lines 1–33 encode sequences of timesteps that have at most $K_{\max}$ steps that are not associated with an observation. A trace is encoded by three predicates:

`time(T)` representing a timestep T;

`succ(T1,T2)` representing the succession between two timesteps $T1$ and $T2$ of a same trace;

`gap(E,S)` representing the length difference S between an input time series E and its associated traces.

Lines 1–19 define the trace lengths, and Lines 21–30 define a bijective mapping between observations and trace timesteps. In particular, Lines 25–26 ensure that the bijective mapping (Lines 28–29) keeps the order of observations. The mapping between an observation T_o and a trace timestep T_s is encoded by the predicate `map(To,Ts)`. Finally, Line 33 ensures that the first observation of each input time series is mapped to the first timestep of the associated trace.

Prior Knowledge Network. The encoding of the prior knowledge network (PKN) is the same as in Appendix B.2 (Lines 35–69).

```
35 % Definition
36 {clause(N,1..C,L,S): in(L,N,S), maxC(N,C), node(N), node(L)}.
37 {clause(N,1..C,L,S)} :- in(L,N,S), maxC(N,C), node(N).
38
39 % Clauses have the smallest valid number possible
40 :- clause(N,C,_,_); not clause(N,C-1,_,_); C > 1.
41
42 % Regulatory functions are monotone
43 :- clause(N,_,L,S), clause(N,_,L,-S).
44
45 % No clause is a subset of another one
```

```
46 :- size(N,C,X), size(N,C',X'), C != C', X <= X',
47     clause(N,C',L,S): clause(N,C,L,S).
48
49 % Nodes that do not have functions are constant
50 1 {constant(N,(-1;1))} 1 :- node(N), not clause(N,_,_,_).
51 constant(N) :- constant(N,_).
52
53 % Sort by length
54 size(N,C,X) :- clause(N,C,_,_), X=#count{L,S: clause(N,C,L,S)}.
55 :- size(N,C,X), size(N,C',X'), C < C', X > X'.
56
57 % Sort by lexicographic order
58 clausediff(N,C,C',L) :- clause(N,C,L,_), not clause(N,C',L,_),
59     clause(N,C',_,_).
60 mindiff(N,C,C',L) :- clausediff(N,C,C',L),
61     L <= L': clausediff(N,C,C',L').
62 :- size(N,C,X), size(N,C',X), C < C',
63     mindiff(N,C,C',L), mindiff(N,C',C,L'), L > L'.
64
65 % External metabolites are disabled by default
66 constant(A,-1) :- ext(A).
67
68 % Constant functions are prohibited
69 :- constant(A), not ext(A).
70
71 % Reversible reactions have the same regulatory rules
72 node(Rr) :- rev(Rf,Rr), node(Rf).
73 node(Rf) :- rev(Rf,Rr), node(Rr).
74 clause(Rr,C,L,V) :- rev(Rf,Rr), clause(Rf,C,L,V).
75 clause(Rf,C,L,V) :- rev(Rf,Rr), clause(Rr,C,L,V).
```

The only difference lies in the encoding of rules of reversible reactions. Indeed, in these ASP programs, we encode reversible reactions as two distinct reactions: a forward reaction and a reverse reaction. Lines 72–75 ensure that the forward and reverse counter-part reactions of a reversible reaction have the same regulatory rules.

Breaking symmetry. The DNF encoding of Boolean formulas introduced in Chevalier et al. (2020), and used in the relaxed inference problem encoding, allows inferring semantically equivalent BN, even with conjunctions ordering. For instance, if `clause(N,1,B,1)` is a rule for node `N` and that there exists an edge $(A, N, 1)$ always true in the observation, then `clause(N,1,A,1). clause(N,2,B,1).` will be a valid rule too. However, adding `B` does not provide any information to the rule since it is always true in the observation. Therefore, one may prevent this solution from being generated.

```
77 valid_lit(T,N,C,A) :- clause(N,C,A,V), read(T,A,-V),
```

```
78      read(T,A',V'): clause(N,C,A',V'), A' != A.
79  valid_lit(T,N,C,A) :- read(T,A,_), clause(N,C,A,_),
80      A = A': clause(N,C,A',_).
81  :- not valid_lit(_,N,C,A), clause(N,C,A,_).
82  valid_clause(T,N,C) :- clause(N,C,_,_), eval(T,N,C,1),
83      eval(T,N,C',-1): clause(N,C',_,_), C != C'.
84  valid_clause(T,N,C) :- eval(T,N,C,_), clause(N,C,_,_),
85      C = C': clause(N,C',_,_).
86  :- not valid_clause(_,N,C), clause(N,C,_,_).
87  valid_rule(N,C,A) :- valid_lit(T,N,C,A), valid_clause(T,N,C).
88  :- not valid_rule(N,C,A), clause(N,C,A,_).
```

First, Lines 77–81 ensure that each literal l in a clause affects the clause behavior, that is, without l the behavior of the clause, regarding the traces, will change. Then, Lines 82–86 ensure that each clause in the formula affects the formula output, regarding the traces. Finally, Lines 87–88 ensure that each regulatory rule is only composed of a minimal number of elements to explain the observations.

Boolean transitions. The synchronous dynamics of the Boolean network (BN) is encoded in the same way as in Appendix B.2 (Lines 90–130).

```
90  % Synchronous
91  update(T,A) :- time(T), node(A), not ext(A),
92      not param(mutation,E,A,_), T=(E,_).
93
94  % External metabolites are read as their state at T+1.
95  read(T,A,V) :- ext(A), succ(T,T'), v(T',A,V), in(A,_,_).
96  read(T,A,V) :- ext(A), succ(T,T'), v(T',A,V), in(_,A,_).
97
98  % Genes are read as their state at T+1.
99  read(T,A,V) :- succ(T,T'), v(T',A,V), in(A,_,_).
100 read(T,A,V) :- succ(T,T'), v(T',A,V), in(_,A,_).
101
102 % Clause evaluation
103 eval(T,A,C, 1) :- update(T,A), clause(A,C,_,_),
104     read(T,L,V): clause(A,C,L,V).
105 eval(T,A,C,-1) :- update(T,A), clause(A,C,L,V), read(T,L,-V).
106
107 % Formula evaluation
108 eval(T,A, 1) :- update(T,A), clause(A,C,_,_), eval(T,A,C,1).
109 eval(T,A,-1) :- update(T,A), eval(T,A,C,-1): clause(A,C,_,_).
110 eval(T,A,V)  :- update(T,A), constant(A,V).
111
112 % Value for mutated node
113 x(T',A,V) :- param(mutation,E,A,V), succ(_,T'), T'=(E,_).
114
115 % Value of external metabolites
116 x(T',A,V) :- ext(A), not r(_,_,A), succ(T,T'), read(T,A,V).
```

```
117
118 % Value for not updated nodes
119 x(T',A,V) :- succ(T,T'), not update(T,A), not ext(A),
120     not param(mutation,E,A,_), x(T,A,V), T'=(E,_).
121
122 x(T',A,V) :- not ext(A), succ(T,T'), update(T,A), eval(T,A,V).
123
124 % Reactions that are not node are active by default
125 x(T',A,1) :- succ(_,T'), not node(A), r(_,_,A),
126              not param(mutation,E,A,_), T'=(E,_).
127
128 % Elements that are not nodes are active by default
129 x(T',A,1) :- succ(_,T'), not node(A), in(_,A,_),
130              not param(mutation,E,A,_), T'=(E,_).
```

Boolean metabolic steady-states. The encoding of Boolean metabolic steady-states is the same as in Appendix B.2 (Lines 132–167). The only difference lies in Line 162 which defines the behavior of reversible reactions such that: a reversible reaction is either not active, in the forward direction, or the reverse direction.

```
132 % Internal structure of the metabolic network
133 r(r,A,R) :- reactant(A,R,_), not ext(A).
134 r(p,A,R) :- product(A,R,_), not ext(A).
135
136 % All the elements of the metabolic network
137 varm(A) :- r(_,A,_).
138 varm(R) :- r(_,_,R).
139 varm(A) :- ext(A).
140
141 % All the elements of the regulatory network
142 varx(A) :- in(_,A,_), not varm(A).
143
144 % Binary states associated with each element
145 1 { v(T,A,(-1;1)) } 1 :- time(T), varm(A).
146 1 { v(T,A,(-1;1)) } 1 :- time(T), varx(A).
147
148 % Observations are fixed
149 :- obs(To,A,V), map(To,Ts), v(Ts,A,-V).
150
151 % An active internal metabolite must produced/consumed
152 :- r(S,A,_), v(T,A,1), v(T,R,-1): r(S,A,R).
153
154 % All products/reactants of an active internal reaction are active
155 :- r(_,A,R), v(T,R,1), v(T,A,-1).
156
157 % Import reaction has all its reactants at T
158 :- ext(A), reactant(A,R,_), v(T,R,1), v(T,A,-1).
159
```

```
160 % Reversible reaction can only be activated in one direction
161 :- rev(Rf,Rr), Rf!=-1, Rr!=-1, v(T,Rf,1), v(T,Rr,1).
162
163 % State of regulator depends on the regulatory state
164 :- varx(A), x(T,A,V), v(T,A,-V).
165
166 % Reaction can be inhibited by the regulatory state
167 :- r(_,_,R), node(R), x(T,R,-1), v(T,R,1).
```

External metabolite dynamics. The traces can contain regulatory Boolean metabolic steady-states that are not mapped to observations. Therefore, it is necessary to model the dynamics of external metabolites.

```
169 % New external metabolites should be produced at T-1
170 :- succ(_,T), succ(T,T'), ext(A), v(T,A,-1), v(T',A,1),
171    v(T,R,-1): product(A,R,_).
172
173 % Removed external metabolites should be consumed at T-1
174 :- succ(T,T'), ext(A), v(T,A,1), v(T',A,-1),
175     v(T,R,-1): reactant(A,R,_).
```

In Lines 170–171, we ensure that for an external metabolite to become available at a time t, it should be produced at $t-1$. In the same way, Lines 173–174 ensure that if an external metabolite becomes unavailable at a time t then it should have been consumed at $t-1$.

Optimization. We aim at inferring only the BNs that best fit the input time series, *i.e.* BNs that are associated with compatible traces of minimal length. The optimization criterion is encoded in Line 178 by minimizing the sum of the length differences between input time series and their associated traces.

```
177 % Objective function: find the traces that best fit the observations
178 #minimize{S, E: gap(E,S)}.
```

Display answer sets. Finally, Lines 180–181 allows displaying only the `clause/4` predicates when showing answer sets.

```
180 #show.
181 #show clause/4.
```

B.3.3 Quantified Linear Constraints

In this section, we present the *MerrinASP* encoding of the quantified linear constraints of the hybrid inference problem. It relies on the extended ASP syntax

described in Appendix B.3.1.

Recall that the hybrid inference problem, as formulated in Chapter V, has:

— existentially quantified linear constraints: the regulated Flux Balance Analysis (rFBA) equations with a lower bound on the growth optimum.

— universally quantified linear constraints: the rFBA equations with an upper bound on the growth optimum (*i.e.* all regulated metabolic steady-states should satisfy the upper bound).

MERRIN. Note that in *MERRIN* (Chapter IV), the linear optimization problems are directly built from the ASP atoms during the solving process. It does not rely on the linear constraints encoding described in this section. However, *MERRIN*'s instantiation of linear constraints is based on the same reasoning as the encoding of linear constraints. Atoms in the following rule bodies are captured during the solving process to instantiate the quantified linear constraints.

Existental constraints. In the hybrid inference problem, the existentially quantified linear constraints are the rFBA equations (Lines 1–20).

```
1  % Variables
2  &dom(check(Ts)){L..U} = f(R) :- time(Ts), Ts=(E,_),
3                                  param(transport,E,R,L,U).
4  &dom(check(Ts)){L..U} = f(R) :- time(Ts), map(To,Ts),
5                                  bound(To,R,L,U).
6  &dom(check(Ts)){L..U} = f(R) :- time(Ts), r(_,_,R),
7                                  bound(R,L,U).
8
9  % Steady-state
10 &sum(check(Ts)){S   f(R): reactant(M,R,S);
11                 S   f(R): product(M,R,S)} = 0 :- time(Ts),
12                                                  not ext(M),
13                                                  r(_,M,_).
14
15 % Inhibition due to missing input metabolite in the substrate
16 &sum(check(Ts)){f(R)} = 0 :- time(Ts), ext(M), reactant(M,R,_),
17     v(Ts,M,-1).
18
19 % Inhibition due to regulatory rules
20 &sum(check(Ts)){f(R)} = 0 :- time(Ts), r(_,_,R), x(Ts,R,-1).
21
22 % Forced metabolic flux for reactions of interest
23 &sum(check(Ts)){f(R)}  = 0 :- time(Ts), in(R,_,_), r(_,_,R),
24                               v(Ts,R,-1).
25 &sum(check(Ts)){f(R)} >= E :- time(Ts), in(R,_,_), r(_,_,R),
26                               v(Ts,R, 1), epsilon(E).
```

```
27
28 &sum(check(Ts)){f(Obj)} >= LB    O :- time(Ts), map(To,Ts), obj(To,O),
29                                      objective(Obj), lb(LB).
```

The activity bounds of reactions, and thus the domain of linear variables, are defined in Lines 2–7: Lines 2–3 fix bounds according to experimental constraints, Lines 4–5 fix bounds according to external metabolites availability, and Lines 6–7 fix bounds according to the metabolic networks bounds. The steady-state equations are given in Lines 10–13, while the flux of inhibited reactions is fixed to zero in Lines 16–20.

Lines 23–26 force the activity of observed reactions. It ensures that there exists a metabolic steady-state compatible with the reactions considered *actives* and *inactives* by the regulatory state. These constraints are necessary to handle metabolic feedback to the regulatory network.

Finally, Lines 28–29 fix a lower bound on the optimum value. It ensures that there exists a metabolic steady-state that has a growth phenotype compatible with the observation.

Linear constraint partitioning. Note that the existentially quantified linear constraints are grouped by timesteps Ts with the partition arguments of the linear theory atoms (`check(Ts)`). This allows *MerrinASP* to solve each set of linear equations independently for each timestep.

Universally quantified linear constraints. The following equations are only applied to timesteps mapped to observations. Indeed, these constraints are used to ensure that the observed growth phenotypes match with the estimated ones. Like previously, Lines 30–48 encode for the rFBA equations.

The universal quantifier is represented by Lines 51–52 through the use of `&assert`. It ensures that all linear assignments compatible with the linear constraints defined by Lines 30–48 have a growth lesser or equal to the observed growth value. This constraint captures the controls of the regulatory network on the metabolic network.

```
31 % Variables
32 &dom(reg(To)){L..U} = f(R) :- next(_,To), obj(To,_),
33                               param(transport,E,R,L,U), To=(E,_).
34 &dom(reg(To)){L..U} = f(R) :- next(_,To), obj(To,_), bound(To,R,L,U).
35 &dom(reg(To)){L..U} = f(R) :- next(_,To), obj(To,_), r(_,_,R),
36                               bound(R,L,U).
37
38 % Steady-state
39 &sum(reg(To)){S   f(R): reactant(M,R,S);
40               S   f(R): product(M,R,S)} = 0 :- next(_,To),
41                                                not ext(M), r(_,M,_).
42
43 % Inhibition due to missing input metabolite in the substrate
44 &sum(reg(To)){f(R)} = 0 :- next(_,To), map(To,Ts), ext(M),
45                            reactant(M,R,_), v(Ts,M,-1).
46
47 % Inhibition due to regulatory rules
48 &sum(reg(To)){f(R)} = 0 :- next(_,To), map(To,Ts), r(_,_,R),
49                            x(Ts,R,-1).
50
51 % Ensure that the biomass optimum match the observation
52 &assert(reg(To)){f(Obj)} <= UB   O :- next(_,To), objective(Obj),
53                                       ub(UB), obj(To,O).
```

Like for existentially quantified linear constraints, universally quantified linear constraints are grouped by timesteps To with the partition arguments of the linear theory atoms (`reg(To)`).

C Relaxed Inference Problem: Application to a Core-Carbon Metabolism Model

In this section, we apply the Boolean relaxation of the inference problem and its saturation-based implementation presented in Chapter III on a small-scale model of core-carbon metabolism (Covert et al., 2001). For the rest, this instance will be denoted by *Core model*. The application extends the results of Thuillier et al. (2021), where the relaxed inference problem has been applied to a *toy* model derived from the *core* model.

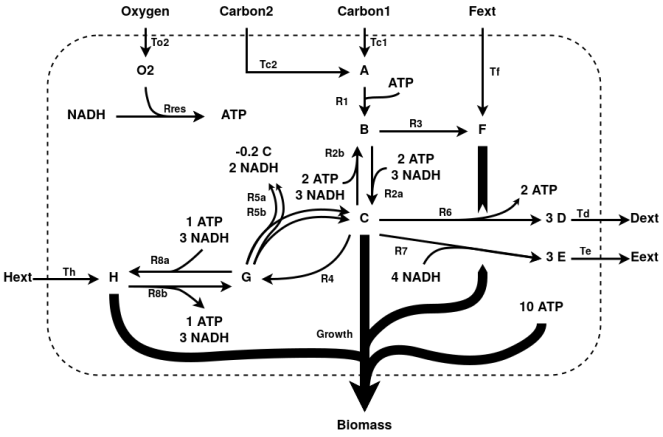
In Section C.1, we describe the instance of the Boolean relaxation of the inference problem for the *core* model. The results are presented in Section C.2.

C.1 Instance Description

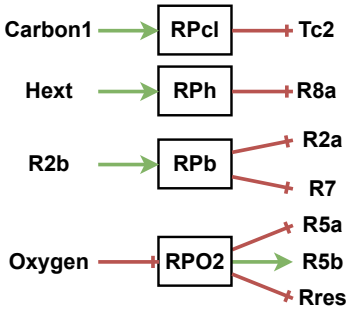
Regulated metabolic network. The *core* model's regulated metabolic network is shown in Fig. 23. Its metabolic network (Fig. 23a) contains 8 external metabolites ($k = 5$ inputs: Carbon1, Carbon2, Oxygen, Fext, Hext; 3 outputs: Biomass, Dext, Eext) and $m = 20$ reactions. Its regulatory system contains $d = 4$ regulatory proteins ({RPcl, RPO2, RPb, RPh}). It is modelled by a Boolean network (Fig. 23c) of dimension $n = k + d + m = 29$. All the functions associated with reactions are set to 1, except for the reactions Tc2, Rres, R2a, R2b, R5a, R5b, R7 and R8a (Fig. 23b).

The *core* model contains a more complex structure than the toy example presented in Thuillier et al. (2021). In particular, the *core* model's metabolic network contains reaction cycles (*e.g.* {R4, R5a} or {R2a, R2b}) whose dynamics are not correctly modeled with our abstraction Boolean of r-dFBA. The non-consideration of stoichiometry can lead to a self-activated cycle leading to spurious Boolean metabolic steady-states.

Experiments. The input time series data were generated from the five experiments studied in Covert et al. (2001). Each experiment is based on a different set $A \subseteq \text{Inp} = \{\text{Carbon1, Carbon2, Oxygen, Fext, Hext}\}$ of initially available input metabolites. The initialization of each experiment is detailed in Tab. 24a. Note that although experiments 3 and 4 appear to be identical, this is not the case. They do have the same two sets of external metabolites available ({Carbon2, Oxygen, Hext}), but their initial concentrations are different: for the experiment 3, they are {Carbon2 = 10, Oxygen = 100, Hext = 2}; for the experiment 4, they are {Carbon2 = 5, Oxygen = 100, Hext = 10}. Thus, both experiments lead to different simulations.



(a) *Metabolic Network*



(b) Influence graph $G(f)$ of the regulatory Boolean network f. Nodes without in-going or out-going edges are not represented. Positive edges are drawn in green with a regular tipping arrow, negative edges are drawn in red with a bar arrow.

	Regulatory proteins				Input metabolites				
Local function	$f_{\text{RPO2}}(x)$	$f_{\text{RPcl}}(x)$	$f_{\text{RPb}}(x)$	$f_{\text{RPh}}(x)$	$f_{\text{Carbon1}}(x)$	$f_{\text{Carbon2}}(x)$	$f_{\text{Oxygen}}(x)$	$f_{\text{Fext}}(x)$	$f_{\text{Hext}}(x)$
Boolean rule	$\neg x_{\text{Oxygen}}$	x_{Carbon1}	x_{R2b}	x_{Hext}	0	0	0	0	0

	Reactions									
Local function	$f_{\text{Tc1}}(x)$	$f_{\text{Tc2}}(x)$	$f_{\text{To2}}(x)$	$f_{\text{Td}}(x)$	$f_{\text{Te}}(x)$	$f_{\text{Tf}}(x)$	$f_{\text{Th}}(x)$	$f_{\text{Growth}}(x)$	$f_{\text{Rres}}(x)$	$f_{\text{R1}}(x)$
Boolean rule	1	$\neg x_{\text{RPcl}}$	1	1	1	1	1	1	$\neg x_{\text{RPO2}}$	1

	Reactions									
Local function	$f_{\text{R2a}}(x)$	$f_{\text{R2b}}(x)$	$f_{\text{R3}}(x)$	$f_{\text{R4}}(x)$	$f_{\text{R5a}}(x)$	$f_{\text{R5b}}(x)$	$f_{\text{R6}}(x)$	$f_{\text{R7}}(x)$	$f_{\text{R8a}}(x)$	$f_{\text{R8b}}(x)$
Boolean rule	$\neg x_{RPb}$	1	1	1	$\neg x_{RPO2}$	x_{RPO2}	1	$\neg x_{RPb}$	$\neg x_{RPh}$	1

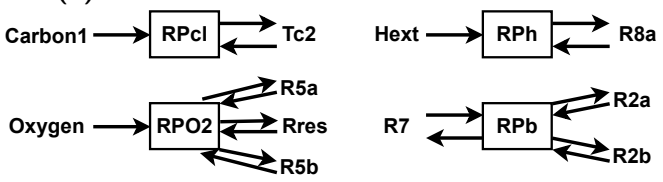
(c) *Boolean Network*. All Boolean functions equal to 1 are reactions that are not regulated by the Boolean network.

■ **Figure 23** – Core-carbon metabolism model introduced in Covert et al. (2001). In the metabolic network (a), each node represents a metabolite, and each hyperedge a reaction. For instance, the hyperedge R7 linking {C; NADH} to {E} models the reaction C + 4 NADH → 3 E. Integer values over hyperedges are stoichiometric coefficients, the default value is 1. (b) shows the influence graph of the Boolean network in (c), with square nodes denoting the regulatory proteins. (c) defines the Boolean network controling the metabolic network in (a), with $x \in \mathbb{B}^{29}$.

Search domain. The search domain $\mathbb{F}$ for the inferred BNs is delimited by the influence graph $\mathcal{G}$ of Fig. 24b. In particular, $\mathbb{F}$ contains all the BNs such that $\forall\, i \in \text{Inp} \cup \mathcal{R} \setminus \{\text{Tc2, Rres, R2a, R2b, R5a, R5b, R7, R8a}\}$, $f_i(x) = 1$, and where f_{RPcl} can depend on Carbon1 and Tc2; f_{RPO2} can depend on Oxygen, Rres, R5a and R5b; f_{RPh} can depend on Hext and R8a; f_{RPb} can depend on R7, R2a and R2b; f_{Tc2} can depend on RPcl; f_{Rres}, f_{R5a} and f_{R5b} can depend on RPO2; f_{R8a} can depend on RPh; and f_{R2a}, f_{R2b} and f_{R7} can depend on RPb. Overall, $\mathbb{F}$ contains 2.9×10^{12} BNs.

	Input Metabolite					Regulatory Protein			
Experiment	$\overline{z_{\text{Carbon1}}}$	$\overline{z_{\text{Carbon2}}}$	$\overline{z_{\text{Oxygen}}}$	$\overline{z_{\text{Fext}}}$	$\overline{z_{\text{Hext}}}$	x_{RPcl}	x_{RPO2}	x_{RPb}	x_{RPh}
1	1	1	1	0	0	1	0	0	0
2	0	1	1	0	0	0	0	0	0
3	0	1	1	0	1	0	0	0	1
4	0	1	1	0	1	0	0	0	1
5	1	1	1	1	1	1	0	0	1

(a) Initial states of the five rFBA simulations.



(b) Influence graph $\mathcal{G}$ delimiting the domain of putative BNs $\mathbb{F}$. Nodes without in-going or out-going edges are not represented. Black regular tipping arrows are unsigned edges, *i.e.* both positive and negative edges.

■ **Figure 24** – Input data for the *core* model. Tab. (a) summarizes the experimental conditions used to generate the input simulations. Fig. (b) shows the influence graph delimiting the search domain for the inference problem.

Simulations. For each one of the five experiments, a rFBA simulation has been run using FlexFlux (Marmiesse et al., 2015). The rFBA simulations are shown in Fig. 25. Each simulation has 200 metabolic steady-states. The regulatory proteins were initialized according to the initial value of external metabolites, *i.e.* $x_{\text{RPcl}} = \overline{z_{\text{Carbon1}}}$, $x_{\text{RPO2}} = \neg\overline{z_{\text{Oxygen}}}$, $x_{\text{RPb}} = 0$ and $x_{\text{RPh}} = \overline{z_{\text{Hext}}}$ where the external metabolite values are given by $(\overline{z_{\text{Carbon1}}}, \overline{z_{\text{Carbon2}}}, \overline{z_{\text{Oxygen}}}, \overline{z_{\text{Fext}}}, \overline{z_{\text{Hext}}})$ (Tab. 24a). The simulations were then binarized as detailed in Chapter III (Tab. 13).

Boolean objective function. To solve the inference problem, one must supply a Boolean objective function $\hat{o}$. Given the set of input metabolites Inp = {Carbon1, Carbon2, Oxygen, Fext, Hext} and the set of output metabolites Out = {Biomass, Dext, Eext}, the objective function was defined as:

$$\forall\, x \in \text{MSS}^{\mathbb{B}}(\mathcal{N}),\ \hat{o}(x) = \sum_{e \in \text{Inp}\setminus\{\text{Oxygen}\}} x_{\text{e}} + \sum_{e \in \text{Out}} x_{\text{e}}$$

This function was motivated by the fact that the maximization of biomass production often corresponds to the maximization of inputs according to the steady-state constraints. In our case, we could not use Oxygen in our score function, since it could lead to spurious Boolean metabolic steady-states.

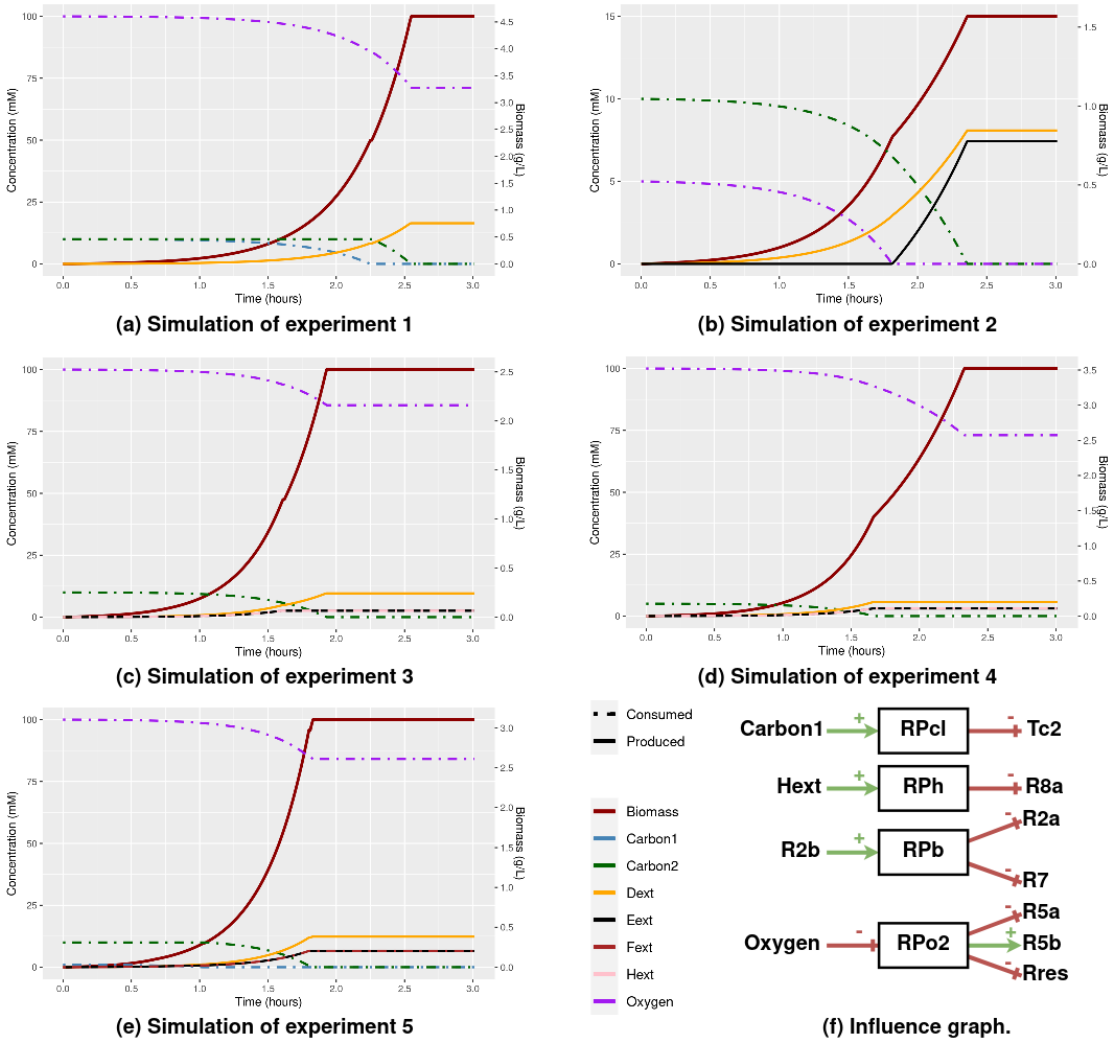


Figure 25 – Simulations of the *core* regulated metabolic model described in Fig. 23 for each experiment (Fig. 24a). Simulations are made with *FlexFLux* with a timestep set to 0.01h. Reaction domains are $\forall r \in \{\text{Tc1, Tc2}\}$, $(l_r, u_r) = (0, 10.5)$, $\forall r \in \{\text{Td, Te}\}$, $(l_r, u_r) = (0, 12.0)$, $\forall r \in \{\text{Te, Tf}\}$, $(l_r, u_r) = (0, 5.0)$, $\forall r \in \{\text{R1, R2a, R2b, R3, R4, R5a, R5b, R6, R7, R8a, R8b, Rres, Growth}\}$, $(l_r, u_r) = (0, 9999)$ and $(l_{\text{To2}}, u_{\text{To2}}) = (0, 15.0)$. **(f)** Influence graph of the regulatory network shown in Fig. 23c. Nodes without in-going or out-going edges are not represented. Positive edges are drawn in green with a regular tipping arrow, negative edges are drawn in red with a bar arrow.

		External metabolites					Regulatory proteins			
Experiment	Time	$\overline{z_{\text{Carbon1}}}$	$\overline{z_{\text{Carbon2}}}$	$\overline{z_{\text{Oxygen}}}$	$\overline{z_{\text{Fext}}}$	$\overline{z_{\text{Hext}}}$	$\overline{x_{\text{RPcl}}}$	$\overline{x_{\text{RPO2}}}$	$\overline{x_{\text{RPb}}}$	$\overline{x_{\text{RPh}}}$
1	0	1	1	1	0	0	1	0	0	0
	1	1	1	1	0	0	1	0	0	0
	225	0	1	1	0	0	0	0	0	0
	227	0	1	1	0	0	0	0	0	0
	256	0	0	1	0	0	0	0	0	0
2	0	0	1	1	0	0	0	0	0	0
	1	0	1	1	0	0	0	0	0	0
	183	0	1	0	0	0	0	1	0	0
	237	0	0	0	0	0	0	1	0	0
3	0	0	1	1	0	1	0	0	0	1
	1	0	1	1	0	1	0	0	0	1
	162	0	1	1	0	0	0	0	0	0
	163	0	1	1	0	0	0	0	0	0
	194	0	0	1	0	0	0	0	0	0
4	0	0	1	1	0	1	0	0	0	1
	1	0	1	1	0	1	0	0	0	1
	168	0	0	1	0	1	0	0	0	1
	169	0	0	1	0	1	0	0	1	1
	234	0	0	1	0	0	0	0	1	0
	235	0	0	1	0	0	0	0	0	0
5	0	1	1	1	1	1	1	0	0	1
	1	1	1	1	1	1	1	0	0	1
	69	1	1	1	0	1	1	0	0	1
	104	0	1	1	0	1	0	0	0	1
	105	0	1	1	0	1	0	0	1	1
	106	0	1	1	0	1	0	0	0	1
	107	0	1	1	0	1	0	0	0	1
	182	0	1	1	0	0	0	0	0	0
	183	0	1	1	0	0	0	0	0	0
	185	0	0	1	0	0	0	0	0	0

		Reactions																		
Experiment	Time	$\overline{v_{\text{Tc1}}}$	$\overline{v_{\text{Tc2}}}$	$\overline{v_{\text{To2}}}$	$\overline{v_{\text{Td}}}$	$\overline{v_{\text{Te}}}$	$\overline{v_{\text{Tf}}}$	$\overline{v_{\text{Th}}}$	$\overline{v_{\text{Growth}}}$	$\overline{v_{\text{Rres}}}$	$\overline{v_{\text{R1}}}$	$\overline{v_{\text{R2a}}}$	$\overline{v_{\text{R2b}}}$	$\overline{v_{\text{R3}}}$	$\overline{v_{\text{R4}}}$	$\overline{v_{\text{R5a}}}$	$\overline{v_{\text{R5b}}}$	$\overline{v_{\text{R6}}}$	$\overline{v_{\text{R7}}}$	$\overline{v_{\text{R8a}}}$	$\overline{v_{\text{R8b}}}$
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	1	1	0	1	1	0	0	0	1	1	1	1	0	1	1	1	0	1	0	1	0
	225	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	227	0	1	1	1	0	0	0	1	1	1	1	0	1	1	1	0	1	0	1	0
	256	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	1	0	1	1	1	0	0	0	1	1	1	1	0	1	1	1	0	1	0	1	0
	183	0	1	0	1	1	0	0	1	0	1	1	0	1	1	0	0	1	1	1	0
	237	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	1	0	1	1	1	1	0	1	1	1	1	1	0	1	1	1	0	1	1	0	1
	162	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	163	0	1	1	1	0	0	0	1	1	1	1	0	1	1	1	0	1	0	1	0
	194	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	1	0	1	1	1	1	0	1	1	1	1	1	0	1	1	1	0	1	1	0	1
	168	0	0	1	0	0	0	1	1	1	0	0	1	1	1	1	0	0	0	0	1
	169	0	0	1	0	0	0	1	1	1	0	0	1	1	1	1	0	0	0	0	1
	234	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	235	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	1	1	0	1	1	1	1	1	1	1	1	1	0	0	1	1	0	1	1	0	1
	69	1	0	1	1	1	0	1	1	1	1	1	0	1	1	1	0	1	1	0	1
	104	0	0	1	0	0	0	1	1	1	0	0	1	1	1	1	0	0	0	0	1
	105	0	1	1	1	1	0	1	1	1	1	1	0	1	1	1	0	1	1	0	1
	106	0	1	1	1	0	0	1	1	1	1	0	0	1	1	1	0	1	0	0	1
	107	0	1	1	1	1	0	1	1	1	1	1	0	1	1	1	0	1	1	0	1
	182	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	183	0	1	1	1	0	0	0	1	1	1	1	0	1	1	1	0	1	0	1	0
	185	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 13 – The binarized metabolic steady-states of each experiment are used as input data of the relaxed inference problem.

	$f_{\text{RPO2}}(x)$	$f_{\text{RPcl}}(x)$	$f_{\text{RPb}}(x)$	$f_{\text{RPh}}(x)$	$f_{\text{Tc2}}(x)$	$f_{\text{R7}}(x)$	$f_{\text{R8a}}(x)$
Model 1	$\neg x_{\text{Oxygen}}$	x_{Carbon1}	x_{R2b}	x_{Hext}	$\neg x_{\text{RPcl}}$	$\neg x_{\text{RPb}}$	$\neg x_{\text{RPh}}$
Model 2	$\neg x_{\text{Oxygen}}$	x_{Carbon1}	x_{R2b}	x_{Hext}	$\neg x_{\text{RPcl}}$	$\neg x_{\text{RPb}}$	$\neg x_{\text{RPh}}$
Model 3	$\neg x_{\text{Oxygen}}$	x_{Carbon1}	x_{R2b}	x_{Hext}	$\neg x_{\text{RPcl}}$	$\neg x_{\text{RPb}}$	$\neg x_{\text{RPh}}$

(a) Regulations common to the models.

	$f_{\text{Rres}}(x)$	$f_{\text{R2a}}(x)$	$f_{\text{R2b}}(x)$	$f_{\text{R5a}}(x)$	$f_{\text{R5b}}(x)$	Subset minimal	Ground truth
Model 1	1	1	$\neg x_{\text{R2b}}$	1	1	✓	
Model 2	1	$\neg x_{\text{R2b}}$	1	1	1	✓	
Model 3	$\neg x_{\text{Rres}}$	$\neg x_{\text{RPb}}$	1	$\neg x_{\text{RPO2}}$	x_{RPO2}		✓

(b) Regulations differing between models.

Table 14 – Three inferred BNs for the instance of *core* model described in Fig. C.1. Not shown local functions are set to 1.

Moreover, this objective function takes into account the output metabolites. As shown in the input PKN, the reaction R7 can be regulated or used in a regulation. This reaction leads to the production of Eext (an output metabolite): Eext is produced if and only if R7 is activated. Thus, it seems to be a good assumption to maximize the outputs for this case study.

C.2 Results

We apply our saturation-based implementation of the relaxed inference problem. The goal was to retrieve the ground truth regulatory networks used for the simulations from the discretized r-dFBA simulations.

For the *core* model, 7 680 BNs were inferred, of which 2 are subset minimal (models 1 and 2 in Tab. 14) and one corresponds to the ground truth (model 3 in Tab. 14). These two subset minimal models are identical for all the regulatory proteins: $f_{\text{RPO2}}(x) = \neg x_{\text{Oxygen}}$, $f_{\text{RPcl}}(x) = x_{\text{Carbon1}}$, $f_{\text{RPb}}(x) = \neg x_{\text{R2b}}$, $f_{\text{RPh}}(x) = x_{\text{Hext}}$; and almost all reactions: $f_{\text{Tc2}}(x) = \neg x_{\text{RPcl}}$, $f_{\text{Rres}}(x) = f_{\text{5a}}(x) = f_{\text{5b}}(x) = 1$, $f_{\text{R7}}(x) = \neg x_{\text{RPb}}$, $f_{\text{R8a}}(x) = \neg x_{\text{RPh}}$. The only difference between these two models is in the two reactions R2a and R2b. For *model 1*, there are $f_{\text{R2a}}(x) = 1$ and $f_{\text{R2b}}(x) = \neg x_{\text{RPb}}$. For *model 2*, there are $f_{\text{R2a}}(x) = \neg x_{\text{RPb}}$ and $f_{\text{R2b}}(x) = 1$.

The ground truth model does not correspond to any of the two subset minimal models. These networks differ from the ground truth model by 5 regulations for *model 1* ($\{f_{\text{Rres}}(x)$, $f_{\text{R2a}}(x)$, $f_{\text{R2b}}(x)$, $f_{\text{R5a}}(x)$, $f_{\text{R5b}}(x)\}$) and by 3 regulations for *model 2* ($\{f_{\text{Rres}}(x)$, $f_{\text{R5a}}(x)$, $f_{\text{R5b}}(x)\}$). For the regulations $f_{\text{Rres}}(x)$, $f_{\text{R5a}}(x)$ and $f_{\text{R5b}}(x)$ no regulations were inferred, they are set to 1.

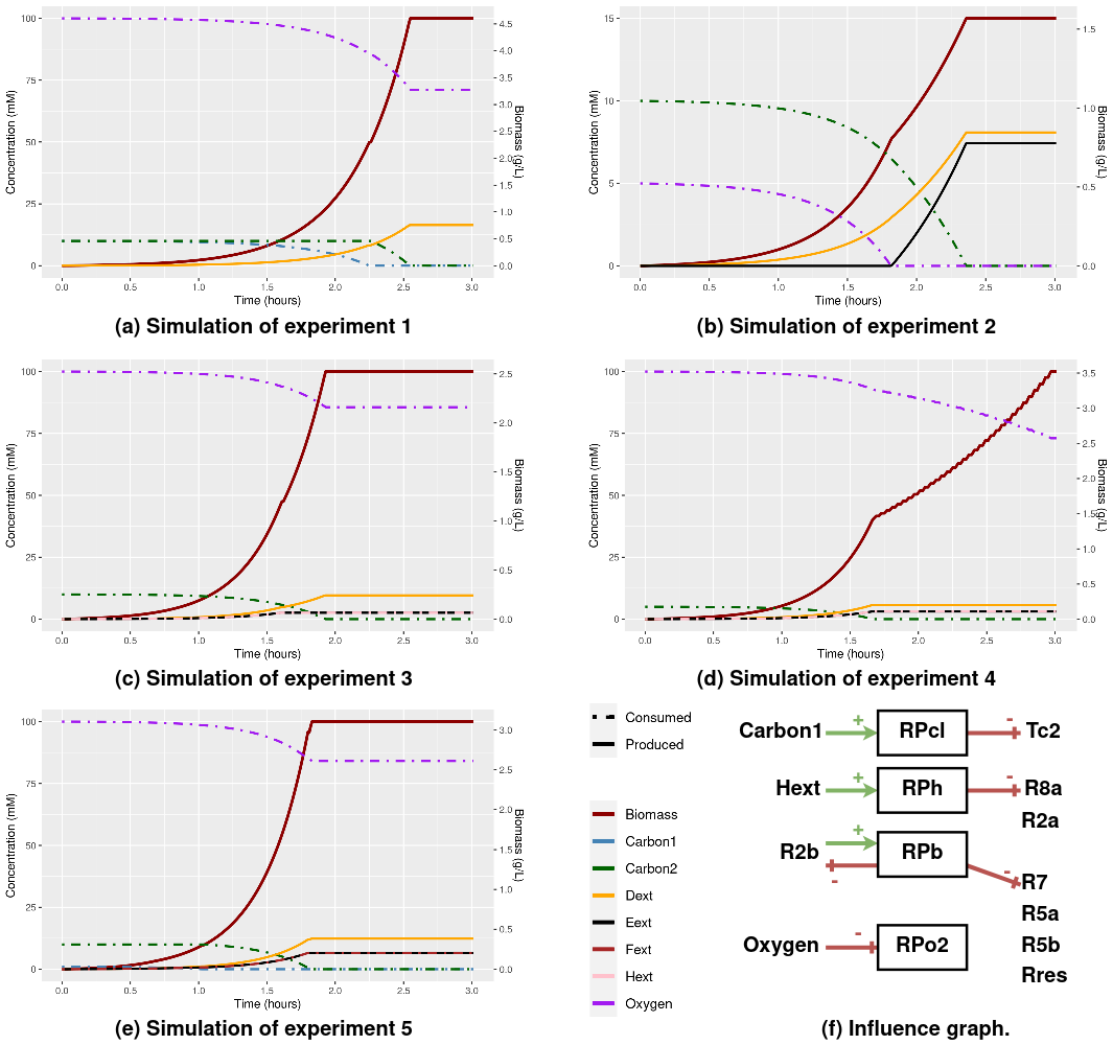
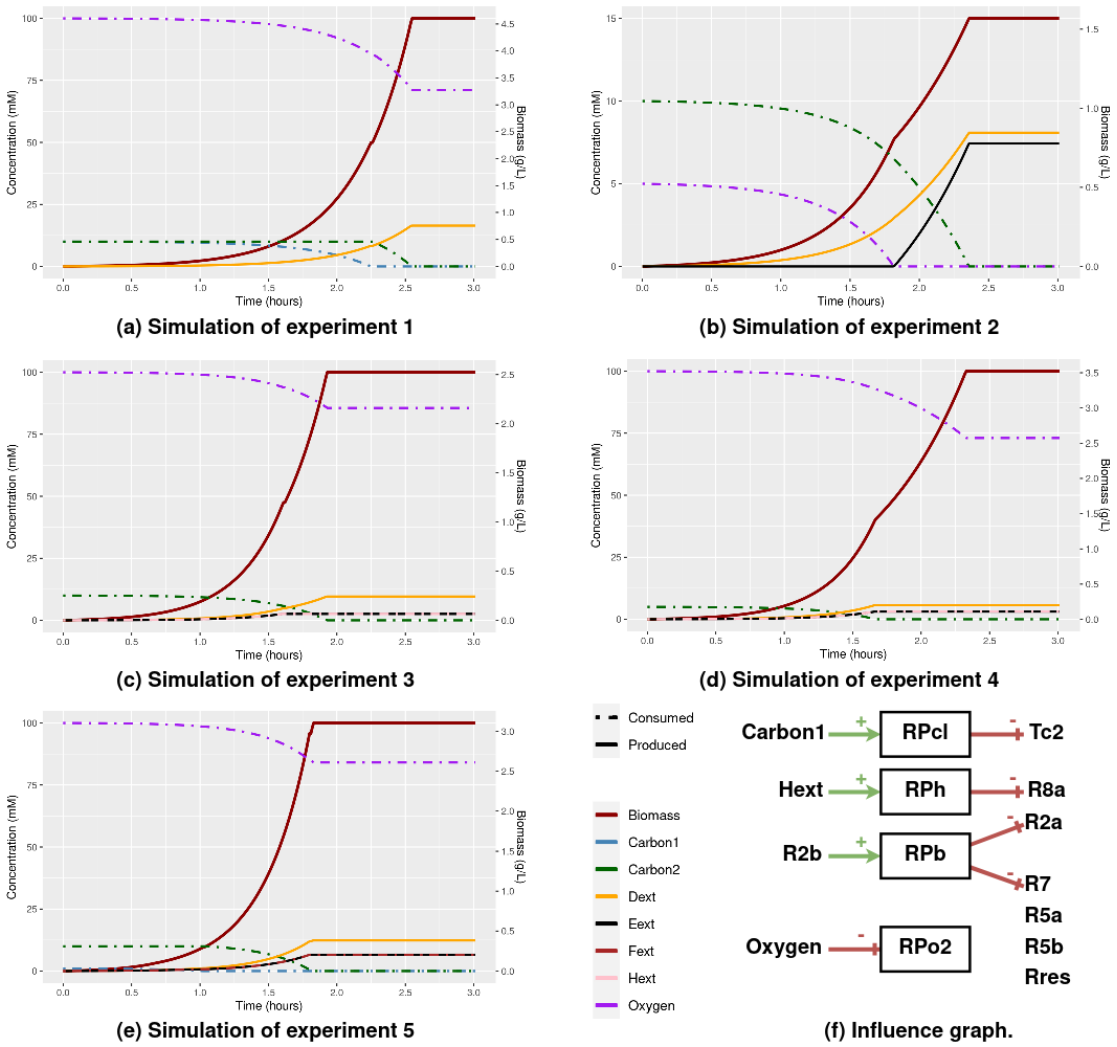


Figure 26 – Simulations of the inferred regulated metabolic network *model 1* (Tab. 14) for each experiment (Tab. 24a). The simulations are made with *FlexFlux* with identical parameters as for Fig. 25. **(f)** Influence graph of the inferred Boolean network *model 1* described in Fig. 14.

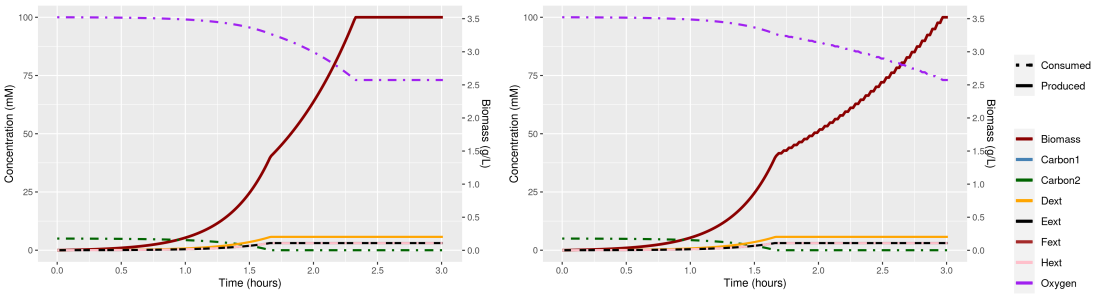
Validation. To check whether the regulated metabolic models inferred could be considered alternatives to the ground truth model, we performed rFBA simulations. In other words, we re-simulate the five experiments with each inferred subset-minimal model. The simulation graphs of the inferred subset minimal model are shown in Figures 26 and 27.

For *model 1*, we observed that the quantitative time-series simulations do not match the input simulations. This model allows reproducing 4 of the 5 input



■ **Figure 27** – Simulations of the inferred regulated metabolic network *model 2* (Tab. 14) for each experiment (Tab. 24a). The simulations are made with *FlexFlux* with identical parameters as for Fig. 25.
(f) Influence graph of the inferred Boolean network *model 2* described in Fig. 14.

simulations (simulation graphs of experiments 1, 2, 3, 5). However, it does not reproduce the simulation of the experiment 4. A comparison between the input simulation of experiment 4 and the simulation issued from *model 1* is given in Fig. 28. In Fig. 28b, simulation of *model 1*, we can see that the production of Biomass is very jerky from time 1.68h, which is not the case in the input simulation (Fig. 28a). The associated binarized metabolic steady-states are shown in Tab. 15. The 5 binarized metabolic steady-states of the timesteps 168, 169, 170, 171, and 172



(a) Ground truth simulation graph **(b)** Model 1 simulation graph

Figure 28 – Simulation graphs of experiment 4 (Fig. 24)comparison between the *ground truth* model and *inferred subset minimal* models. (Tab. 14). (a) is the simulation graph of the ground truth regulated metabolic network (Fig. 23). The simulation graph of the regulated metabolic network controlled by the inferred BN *model 1* (Tab. 14) is identical. (b) is the simulation graph of the regulated metabolic network controlled by *model 2* (Tab. 14).

are repeated until the end of the simulation. These qualitative behaviors induced by *model 1* from time 1.68h do not match with the qualitative behavior of the ground truth model which is composed of only 1 binarized metabolic steady-state. Moreover, during the metabolic steady-states associated with the timesteps 170 and 171 no Biomass is produced. Thus, the cell goes through a series of start-stop phases that do not correspond to any real biological behavior. Note that *model 2* can perfectly reproduce the 5 input simulations. Thus, one of the inferred subset-minimal models could not match the observations. This result shows that our Boolean abstraction of the inference problem is not perfect. It does not capture all the subtleties of the linear dynamics of regulated metabolic systems.

For the subset minimal *model 2*, we observed that these quantitative time-series simulations were strictly identical to the simulations of the toy example used to generate the dataset. This suggests that the regulations on Rres, R5a, and R5b are not necessary to explain the dataset. The inferred models contain all the needed regulations and can be considered as the simplest regulated metabolic models matching with the experimental conditions of Tab. 24a. Already in Covert et al. (2001), the authors recognize that, unlike other regulations, Rres 'regulation is not necessary for the solution' and that R5a and R5b regulations 'are equivalent stoichiometrically' so 'FBA alone would fail to predict which are active under given condition'. These regulations are biologically present only to ensure that unnecessary enzymes decay. However, since enzyme concentrations are not explicitly represented in the rFBA framework, the dataset does not reflect this biological behavior, making it impossible to infer these regulations properly.

	External metabolites					Regulatory proteins			
Time	$\overline{z_{\text{Carbon1}}}$	$\overline{z_{\text{Carbon2}}}$	$\overline{z_{\text{Oxygen}}}$	$\overline{z_{\text{Fext}}}$	$\overline{z_{\text{Hext}}}$	$\overline{x_{\text{RPcl}}}$	$\overline{x_{\text{RPO2}}}$	$\overline{x_{\text{RPb}}}$	$\overline{x_{\text{RPh}}}$
0	0	1	1	0	1	0	0	0	1
1	0	1	1	0	1	0	0	0	1
168	0	0	1	0	1	0	0	0	1
169	0	0	1	0	1	0	0	1	1
170	0	0	1	0	1	0	0	1	1
171	0	0	1	0	1	0	0	0	1
172	0	0	1	0	1	0	0	0	1
173	0	0	1	0	1	0	0	0	1
174	0	0	1	0	1	0	0	1	1
175	0	0	1	0	1	0	0	1	1
176	0	0	1	0	1	0	0	0	1
177	0	0	1	0	1	0	0	0	1
...									
299	0	0	1	0	0	0	0	0	0

	Reactions																			
Time	$\overline{v_{\text{Tc1}}}$	$\overline{v_{\text{Tc2}}}$	$\overline{v_{\text{To2}}}$	$\overline{v_{\text{Td}}}$	$\overline{v_{\text{Te}}}$	$\overline{v_{\text{Tf}}}$	$\overline{v_{\text{Th}}}$	$\overline{v_{\text{Growth}}}$	$\overline{v_{\text{Rres}}}$	$\overline{v_{\text{R1}}}$	$\overline{v_{\text{R2a}}}$	$\overline{v_{\text{R2b}}}$	$\overline{v_{\text{R3}}}$	$\overline{v_{\text{R4}}}$	$\overline{v_{\text{R5a}}}$	$\overline{v_{\text{R5b}}}$	$\overline{v_{\text{R6}}}$	$\overline{v_{\text{R7}}}$	$\overline{v_{\text{R8a}}}$	$\overline{v_{\text{R8b}}}$
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	1	1	1	0	1	1	1	1	1	0	1	1	1	0	1	1	0	1
168	0	0	1	0	0	0	1	1	1	0	0	1	1	1	1	0	0	0	0	1
169	0	0	1	0	0	0	1	1	1	0	0	1	1	1	1	0	0	0	0	1
170	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
171	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
172	0	0	1	0	0	0	1	1	1	0	0	1	1	1	1	0	0	0	0	1
173	0	0	1	0	0	0	1	1	1	0	0	1	1	1	1	0	0	0	0	1
174	0	0	1	0	0	0	1	1	1	0	0	1	1	1	1	0	0	0	0	1
175	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
176	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
177	0	0	1	0	0	0	1	1	1	0	0	1	1	1	1	0	0	0	0	1
...																				
299	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table 15 – Binarized metabolic steady-states of experiment 4 from the simulation of regulated metabolic network controlled by the inferred BN *model 1* (Section 14). The 5 same metabolic steady-states (168, 169, 170, 171, and 172) are repeated until the end of the simulation.

COLLEGE DOCTORAL BRETAGNE
MATHS, TELECOMS
INFORMATIQUE, SIGNAL
SYSTEMES, ELECTRONIQUE

Université de Rennes

Titre : Méthodes de Satisfiabilité Hybrides pour l'Inférence de Régulations Booléennes Contrôlant des Réseaux Métaboliques

Mots clés : biologie des systèmes – système dynamique hybride – synthèse de modèles – réseau métabolique régulé – optimisation combinatoire et linéaire – programmation logique

Résumé : Les systèmes biologiques sont des systèmes multi-échelles complexes composés de nombreux mécanismes biologiques interconnectés. Parmi ces échelles, il y a le métabolisme, qui transforme les nutriments en énergie et en biomasse, et le système de régulation, qui agit comme un contrôleur de l'activité métabolique. Modéliser le couplage du métabolisme et de la régulation est difficile et nécessite d'intégrer les formalismes algébriques différentiels modélisant le métabolisme avec les formalismes discrets modélisant la régulation. Bien qu'il existe des formalismes de simulation de la dynamique hybride de ce couplage, il n'existe aucune méthode pour synthétiser les contrôleurs régulant l'activité métabolique, *i.e.* les règles de régulation. Cette thèse présente trois formulations du problème de synthèse comme des problèmes d'optimisation combinatoire sous contraintes, logiques et hybrides (logiques et linéaires), quantifiées. Chaque formulation fait l'objet d'une approche de résolution dédiée. La première repose sur des méthodes de satisfiabilité, tandis que les deux autres utilisent des méthodes de résolution hybrides couplant des contraintes logiques et linéaires. En particulier, la thèse présente une méthode générique pour résoudre les problèmes d'optimisation combinatoire sous contraintes linéaires quantifiées. Ces travaux ont conduit au développement de deux logiciels, *Merrin* et *MerrinASP*, qui étendent le paradigme de programmation par ensembles réponses (ASP) avec des contraintes linéaires quantifiées. Cette thèse met également à disposition des jeux de données synthétiques simulant différents types de données omiques, ainsi que le protocole utilisé pour les générer.

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

Keywords: systems biology – hybrid dynamic system – model synthesis – regulated metabolic network – combinatorial and linear optimization – logic programming

Abstract: Biological systems are complex multi-scale systems composed of many interconnected biological mechanisms. These scales include the metabolism, which transforms nutrients into energy and biomass, and the regulatory system, which acts as a controller of metabolic activity. Modeling the coupling of metabolism and regulation is difficult and requires integrating the differential-algebraic formalisms used to model the metabolism with the discrete formalisms used to model the regulation. Although formalisms for simulating the hybrid dynamics of this coupling exist, no method allows for the synthesis of the controllers that regulate metabolic activity, that is, the regulatory rules. This thesis presents three formulations of the synthesis problem as combinatorial optimization problems under logical and hybrid (logical and linear) quantified constraints. A dedicated solving method is given for each formulation. The first formulation is solved using satisfiability methods, while the other two rely on hybrid solving methods that integrate logical and linear constraints. In particular, the thesis presents a generic framework for solving combinatorial optimization problems under quantified linear constraints. These formalizations have led to two tools, *Merrin* and *MerrinASP*, which extend the answer set programming (ASP) paradigm with quantified linear constraints. This thesis also provides synthetic datasets that simulate different types of omics data, as well as the protocol used to generate them.