Linear Programming for Metabolic Network Completion

Kerian Thuillier
Univ Rennes, Inria, CNRS, IRISA
ENS Rennes DIT student – Dyliss intern
F-35000 Rennes, France
kerian.thuillier@ens-rennes.fr

***Abstract*—Metabolic networks are a helpful tool to represent cells. They contain information about every reaction occurring inside an organism. However, metabolic networks of poorly studied species are incomplete. It is possible to complete these networks with knowledge of other well-known species.**

In this paper, we present a new linear programming approach for the problem of topological activation in metabolic networks based on flows and MTZ.

We developed our tool *Flutampl* with AMPL. *Flutampl* completes draft networks thanks to the hybrid completion method as formulated by *C. Frioux et al* [4]. It returns optimal solutions for the hybrid completion directly from *sbml* files, data format used for modelling metabolic networks.

***Index Terms*—linear programming, MILP, gap-filling, bioinformatics, metabolic networks**

Introduction

Cells are the foundation of living organisms. Among all the tools used to study them, metabolic networks are a way to represent all the chemical reactions used by the cells to transform nutrients into energy and compounds. Such networks are useful to understand the cellular functioning and to determine an efficient way to combine them to produce target compounds.

Although we have a lot of information for well-studied species, which therefore are associated with high-quality metabolic networks, poorly studied organisms suffer from a data incompleteness. To fulfill this lack of knowledge, computational methods were developed to suggest putative reactions allowing the completion of less studied species metabolic networks thanks to reactions appearing in networks of related well-known species. There are many different network completion problems, which can be classified in three classes [11]. The first class of problem is usually solved with linear programming approaches. The second class is formulated as combinatorial problem currently processed with logical programming. The third class is a hybrid version merging combinatorial and linear constraints currently processed with hybrid solver combining linear programming and boolean satisfiability kind of constraints [4].

In this paper, we introduce a new approach to model with linear constraints the topological metabolic network completion problem in order to solve the hybrid problem with Mixed Integer Linear Programming approach (MILP). This problem belongs to the second class of problems and is currently solved with combinatorial approach. The hybrid completion is naturally deduced from the topological completion.

First, we will define what a metabolic network is and describe the hybrid completion problem. Then, we will introduce our solution to implement the combinatorial part in linear programming and explain how we optimise it. Finally, we will present our tool *Flutampl* which solves this problem.

I. Metabolic Network

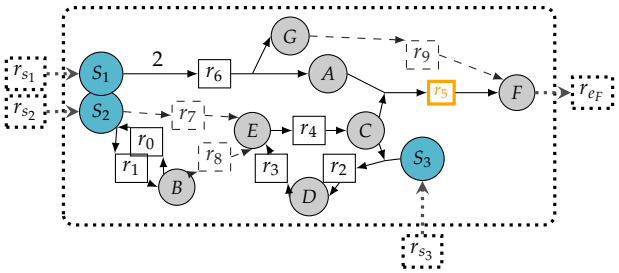

Figure 1. Metabolic network example [4]. Round nodes are metabolites and square nodes are reactions. Blue nodes represent the seeds, the reaction is the target reaction and dotted reactions are import reactions. There are two kinds of nodes: straight nodes are from the draft network while dashed nodes are from the model network. Numbers on arrows are the stoichiometric coefficient, default value is 1.

A. Definitions

A *metabolic network* is a labelled directed bipartite graph $G = (R \cup M, E, s)$ representing all the chemical reactions occurring in an organism. A bipartite graph is a graph whose nodes can be decomposed into two disjoint sets such that two vertices of the same set are not connected by any edge. An example of metabolic network is shown in Figure 1. Sets of nodes R and M represent respectively *reactions* and *metabolite* compounds present in the cell. The set E contains edges of the form $(m, r) \in E$ and $(r, m) \in E$, $m \in M$ and $r \in R$. In the first case, m is called a *reactant* of r, and in the second it is its *product*. Functions $prdts(r)$ and $rcts(r)$ return respectively the sets of products, reactants of a reaction r. A reaction can have multiple ingoing and outgoing edges, *i.e.* a reaction could need several compounds to be initialised and could produce several compounds. In the sequel $s : E \to \mathbb{R}$ is a function associating a stoichiometric coefficient to an edge. Stoichiometric coefficients are the number of compounds consumed or produced by a reaction. Thus, metabolic flux occurring in each reaction are represented by v_r. These flux have bounds defined by two functions $lb : R \to \mathbb{R}$ and $ub : R \to \mathbb{R}$, returning respectively the lower bound and the upper bound for a reaction r. Moreover, we distinguish a subset of metabolites $S \subseteq M$ as the seeds, the network's entries. These are compounds present in the cell's external environment, and then they will be considered present in the network at the system initialisation state. The *system initialisation state* is the state in which none of the reactions are activated and none of the compounds, except the seeds, are activated in the system.

B. Notion of Activation

Before speaking strictly about the problem of metabolic networks completion we must start with the notion of activation.

1) Stoichiometric Activation: A first kind of activation to solve this problem is the stoichiometric activation, and is based on *Flux Balance Analysis* (**FBA**). It is the analysis of the cell's metabolic flux at a steady state. In fact, any chemical reaction requires mass conservation and the non-accumulation of metabolites [12]. A reaction cannot create or destroy matters and an organism does not accumulate compounds; everything must be consumed by reactions.

It is purely a flow problem which is commonly solved by the linear way. It is a basic linear problem in this domain. *CobraPy*, a Python library, implements *FBA* with CPLEX, a linear solvers [6].

$$\forall m \in M,\quad 0 = \sum_{\substack{r \in R \\ m \in prdts(r)}} s(r,m) \times v_r - \sum_{\substack{r \in R \\ m \in rcts(r)}} s(r,m) \times v_r \quad (1)$$

$$\forall r \in R,\ lb(r) \leq v_r \leq ub(r) \quad (2)$$

The equation (1) expresses the non-accumulation of compounds in the cells. Each metabolite is produced as well as consumed. While the equation (2) gives lower and upper bounds to the metabolic flux.

A reaction t is *stoichiometrically activated* in a network G if and only if both equations (1) and (2) are respected and $v_t > 0$.

In previous example Figure 1, r_0 and r_1 are stoichiometrically activated, as the self-activated cycle r_2, r_3, r_4. In fact, r_4 needs to have a flux if r_2 has got one. A positive flux is flowed in this cycle. In contrast, r_6 has got a null metabolic flux because no reaction consumes the metabolite G.

Notice the self-activation of cycles. Thus, it is not possible to be sure that there is a way to reach the steady state from the system initialisation state.

2) Topological Activation: A second activation is the topological one. While the previous method is based on flux, this one is based on topology.

Biologically, a reaction occurs only if all its reactants are disponible, and metabolites are disponible in the cell's environment only if they are produced by at least one reaction. A reaction is *topologically activated* from the seeds only if all its reactant metabolites are also topologically activated. A metabolite is *topologically activated* if and only if it is the product of at least one topologically activated reaction or it is a seed.

Compared to the stoichiometric activation, topological activation constraints are defined recursively. Currently, checking the topological activation of a reaction is done with combinatorial approach using logical programming tools, *MeneTools* [11] and *fluto* [4]. Those tools allow modelling recursive models. This is not possible to use such recursive models in MILP, and thus a linear programming modelling of the problem has never been done.

In the example Figure 1, topologically activated reactions are r_0, r_1 and r_6. The cycle r_2, r_3 and r_4 is not topologically activated from the seeds. There

is no possible way to topologically activate r_2 from topologically activated metabolites.

This kind of activation disables self-activated cycles [11]. All the activated compounds and reactions are activated from the seeds at the system initialisation state.

3) Hybrid Activation: Finally, a *hybridly activated reaction* in a network G from a set of seeds S is a reaction which is both stoichiometrically and topologically activated from S [4].

Hybridly activated reactions Figure 1 are r_0 and r_1 because there are both stoichiometrically and topologically activated.

This activation merges the properties of both previous ones. Hybridly activated reactions can occur thanks to the existence of a metabolic flux at a steady state and can be initiated from the system initialisation thanks to the topological activation. In other words, self-activated cycles of metabolic flux are enabled at a given time.

C. Metabolic Network Completion Problem

To formalise this problem, we first introduce the *draft network* which is the metabolic network that we want to complete, and the *model network* which is a metabolic network whose reactions will be used to complete the draft network. *Metabolic Network Completion Problem* takes as entry two metabolic networks, a draft network G_d and a model network G_m, a set of seeds $S \subseteq M_d$ and a set of target reactions $T \subseteq R_d$. Target reactions are reactions which we want to set reachable according to both stoichiometric and topological activation. The objective is to find a metabolic network $\widetilde{G} = \left(\widetilde{M} \cup \widetilde{R}, \widetilde{E}, \widetilde{s}\right)$ such that $card(\widetilde{R})$ is minimal and for all $t \in T$, t respects selected activation properties where:

$$\widetilde{R} \subseteq R_d \cup (R_m \setminus R_d)$$
$$\widetilde{M} = M_d \cup \left\{ m \mid r \in \widetilde{R}, m \in prdts(r) \cup rcts(r) \right\}$$
$$\widetilde{E} = (E_d \cup E_m) \cap \left(\widetilde{M} \times \widetilde{R} \cup \widetilde{R} \times \widetilde{M} \right)$$
$$\widetilde{s} : e \in \widetilde{E} \rightarrow \text{if } e \in E_d, s_m(e) \in \mathbb{R} \text{ else } s_d(e) \in \mathbb{R}$$

For example in Figure 1, to stoichiometrically activate r_5 we must add r_9 to fully consume G. We must add r_7 or r_8 in order to topologically activate r_5. We deduce that to hybridly activate r_5, we must add $\{r_9, r_7\}$ or $\{r_9, r_8\}$.

Let us define a binary variable $x_{r'}$ for all $r' \in R'$ a reaction of the model network such that $x_{r'} = 1$ if and only if $r' \in \widetilde{R}$. The objective being to add as few model's reactions as possible, the objective function (3) minimises the sum of all those variables.

$$\sum_{r' \in R_m} x_{r'} \tag{3}$$

D. Stoichiometric Metabolic Network Completion

Stoichiometric Metabolic Network Completion Problem defines a metabolic network $\widetilde{G}$ such that for all $t \in T$, t is stoichiometrically activated. It minimises (3) such that $\widetilde{G}$ respects following constraints (1) and (2).

II. Topological Completion with MILP

For encoding the *Topological Metabolic Network Completion Problem*, we based our approach on mixed integer linear programming paradigm. This paradigm relies upon declarative programming; problems are written in the form of linear objective function under linear constraints [2]. We do not need to explain the way to compute solutions but the solutions properties; linear solvers will find optimal solutions respecting all those properties.

Let us now introduce the formulation of the topological activation with linear constraints.

A. Metabolic Network's Constraints

As explained previously, topological activation is modelled with combinatorial recursive constraints. Let us define two binary variables a_r and a_m, representing respectively the topological activation of a reaction $r \in R$ and a metabolite $m \in M$. For r a reaction, all r's reactants must be topologically activated to topologically activate r (4).

$$\forall r \in R, \forall m \in rcts(r),\ a_r \leq a_m \tag{4}$$

For m a topologically activated metabolite, if it is not a seed then m must be produced by at least one topologically activated reaction (5).

$$\forall m \in M \setminus S,\ a_m \leq \sum_{\substack{r \in R \\ m \in prdts(r)}} a_r \tag{5}$$

Seeds and target reactions variables are activated by default (6).

$$\forall e \in S \cup T,\ a_e = 1 \tag{6}$$

Constraints (4), (5) and (6) induce a retro-propagation of metabolites and reactions topological activation. This propagation starts from the target reactions and tries to reach the seeds. It acts as a binary flow without conservation constraints which topologically activates all the needed elements to go from the flow source (target reactions) to the flow target (seeds).

B. Binary Acyclic Flow

Such binary flows can be cyclic and could produce self-activated cycles.

Notice that a reaction can produce several compounds whose part could be needed to produce target reactions. An example of problematic case is shown in Figure 2 with r_3 ending the cycle and producing the target reaction reactant T. Thus, it is not possible to simply remove all the reaction cycles. We need to topologically activate just some part of reactions products.

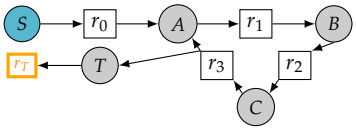

Figure 2. Example of problematic case for cycle removing.

To correctly prohibit flow cycles, we applied a similar method to Miller, Tucker and Zemlin (MTZ). MTZ method consists in labelling the vertices by which a flow passes in a strictly increasing (or decreasing) way [3]. As we want a binary flow which can pass through only part of reactions products, we will modify the MTZ rule such that labels of reactions must be strictly inferior to its products labels only if the edges which link them are topologically activated.

1) Edge Topological Activation: Firstly, we must introduce how to topologically activate an edge. Let us introduce $e_{(r,m)}$ a binary variable representing the topological activation of the edge (r, m) where $r \in R$ and $m \in prdts(r)$. New constraints must be added to the model to define the value of e. For m a metabolite, if m is topologically activated then at least one of its ingoing edges is activated (7).

$$\forall m \in M,\ a_m \leq \sum_{\substack{r \in R \\ m \in prdts(r)}} e_{(r,m)} \tag{7}$$

For r a reaction, if r is topologically activated then at least one of its outgoing edges is activated (8).

$$\forall r \in R,\ a_r \leq \sum_{m \in prdts(r)} e_{(r,m)} \tag{8}$$

For r a reaction, if an outgoing edge of r is topologically activated then r is too (9).

$$\forall r \in R, \forall m \in prdts(r)\ e_{(r,m)} \leq a_r \tag{9}$$

2) Nodes Labelling: Secondly, we need to introduce an upper bound for label values. Since there are at most $|M|$ topologically activated metabolites and $|R|$ topologically activated reactions, then there are in the worst case $|R \cup M|$ different labels. Thus, we can use $|R \cup M|$ as an upper bound for label values. Let us define $l_e \in \mathbb{R}^{+*}$ a variable representing the label value for $e \in R \cup M$. All topologically activated nodes must have a strictly positive label's value while disabled ones have a null label's value (10).

$$\forall e \in R \cup M,\ a_e \leq l_e \leq a_e \times |R \cup M| \tag{10}$$

It is then necessary to define the increased label's value constraints (11, 12).

$$\begin{aligned} &\forall r \in R, \forall m \in rcts(r), \\ &l_m + 1 \leq l_r + (1 - a_r) \times |R \cup M| \end{aligned} \tag{11}$$

$$\begin{aligned} &\forall r \in R, \forall m \in prdts(r), \\ &l_r + 1 \leq l_m + (1 - e_{(r,m)}) \times |R \cup M| \end{aligned} \tag{12}$$

C. Topological Metabolic Network Completion Problem

Topological Metabolic Network Completion Problem defines a metabolic networks $\widetilde{G}$ such that for all $t \in T$, t is topologically activated. It minimises (3) such that $\widetilde{G}$ respects following constraints (4), (5), (6), (7), (8), (9), (10), (11) and (12).

III. Model Implementation

Hybrid Metabolic Network Completion Problem defines a metabolic networks $\widetilde{G}$ such that for all $t \in T$, t is hybridly activated. It minimises (3) such that $\widetilde{G}$ respects following constraints (1), (2), (4), (5), (6), (7), (8), (9), (10), (11) and (12).

A. Model Settings

The linear model is implemented with the proprietary software AMPL based on a mathematic programming language. Open-source alternatives exist as Pyomo [9] or Julia [5] but are less user-friendly. It uses the linear solver CPLEX – a proprietary software developed by IBM.

Using the model on the metabolic networks Figure 1, we get the following output: { reactions to add: r_9, r_7; metabolic flux in r_5: 49999.5 }.

B. Variable Complexity

Linear solvers have limitations. They cannot take too many variables, especially integer ones. In fact, linear problems are polynomial problems if all their variables are reals. Otherwise, it becomes a mixed integer linear problem, and so an NP-Complete problem.

The current implementation of this linear model works perfectly for toy networks, about a few dozen of reactions. However, the computation time explodes for real-sized networks: it does not scale.

This problem is due to a high number of variables. In fact, it has a variable complexity in $O(4|R'| + 3|R| + 3(|M| + |M'|) + |E \cup E'|)$. $|E \cup E'|$ are binary variables representing edges activation.

IV. Model Optimisation

A. Network Rewriting

To reduce the number of used variables, it is possible to rewrite reactions to not have to used topological activation variables on edges. In fact, we can rewrite metabolic network by splitting all multi-products reactions into reactions with the same reactants but only one product. In sequel, we will call sister reactions, all splitted reactions coming from the same one. Let us also define $link(r)$ which return the set of r's sister reaction ($r \in link(r)$)

Formally this gives us a new metabolic network $G_f = \left(R_f \cup M_f, E_f, s_f\right)$ with:

$$
\begin{aligned}
M =& M_f \\
R_f =& \{r_p \mid r \in R, p \in prdts(r)\} \\
E_f =& \{(r_p, p) \mid r \in R, p \in prdts(r)\} \\
&\cup \left\{(p, r) \mid r \in R_f, p \in rcts(r_f)\right\} \\
s_f =& \{((r_p, p), s(r, p)) \mid r \in R, p \in prdts(r)\} \\
&\cup \left\{\left((p, r), \frac{s(p,r)}{link(r)}\right) \mid r \in R_f, p \in rcts(r)\right\}
\end{aligned}
$$

This formatting strategy is used by *A. Julien-Laferrière et al* [1]. We just complete their transformation by upgrading the stoichiometric coefficients. In fact, stoichiometric coefficients of the reaction's reactants are divided by the number of sister reactions. Otherwise it will need more quantity of reactants to produce the same quantity of products. For example, the reaction $A \to B + C$ are divided in $A \to B$ and $A \to C$. It will need $2A$ to produce $B + C$ against $1A$ for the original reaction.

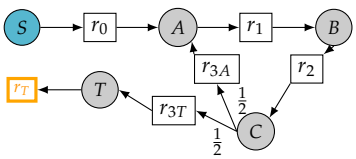

Figure 3. Rewritten version of Figure 2 metabolic network.

Figure 3 is an example of Figure 2 rewriting.

B. Constraint adaptation

As we optimise the input network, we need to update our linear constraints.

1) MTZ *Constraints:* As reactions are already splitted, we could only use reaction topological activation to apply MTZ. It is also possible to reduce the number of variables used as labels, by only labelling metabolites. It is possible to see a metabolic network not as bipartite directed labelled graph but as directed hypergraph [1]. Reactions are hyperarcs having multiple outgoing and ingoing edges. In that case, our flow will only pass through metabolite nodes. It is therefore possible to only label metabolites. Our upper bound will become $|M|$ as it cannot have more than $|M|$ topologically activated metabolites.

So as previously, topologically activated metabolites have a strictly positive label (13) and labels are increasing (14).

$$\forall m \in M,\ a_m \leq l_m \leq a_m \times |M| \tag{13}$$

$$\begin{aligned}&\forall r \in R, \forall m_1 \in rcts(r), \forall m_2 \in prdts(r),\\ &l_{m_1} + 1 \leq l_{m_2} + (1 - a_r) \times |M|\end{aligned} \tag{14}$$

2) Stoichiometric Constraints: For the stoichiometric activation, we just need to add another constraint. In fact, sister reactions must have the same metabolic flux. It is not possible to produce one of them without the others. This additional constraint can be formulated as below (15).

$$\forall r_1 \in R, r_2 \in link(r),\ v_{r_1} = v_{r_2} \tag{15}$$

3) Completion Constraints: The last constraint to add to correctly be able to use the optimisation is a completion constraint. In fact, if we add a reaction to the draft network, we must also add all its sister reactions (16).

$$\forall r_1 \in R', r_2 \in link(r),\ x_{r_1} = x_{r_2} \tag{16}$$

C. Hybrid Metabolic Network Completion Model

This new version of the *Hybrid Metabolic Network Completion Problem* defines a metabolic networks $\widetilde{G}$ such that for all $t \in T$, t is hybridly activated. It minimises (3) such that $\widetilde{G}$ respects following constraints (1, 2, 4, 5, 6, 13, 14, 15 and 16).

It has a number of variables in $O(2|R'| + |R| + 3(|M| + |M'|)$ and can process real-sized metabolic network. See next section for more details.

V. Results

A. Flutampl

With this optimisation, we developed *Flutampl* a linear programming tool processing hybrid completion. This tool can read *sbml* files, a biological file format used to model metabolic networks, as input and return solution in the same format.

Flutampl is stored on a private GitHub repository.

B. Results Verification

To test the model, we used the *Meneco* benchmark of *Escherichia Coli iJR904* [11] with *M_adp_c* as seed. This benchmark is composed of 3 600 networks assembled with our scripts. Those networks were obtained by degrading the full *Escherichia Coli iJR904* metabolic networks by 10, 20, 30 or 40 percent and modifying objective reactions. In order to obtain a benchmark that can be tested on a personal computer, we selected 360 networks : 180 degraded by 10%, and 180 degraded by 40%. A timeout of 1 000 seconds is fixed.

Flutampl's results are verified with *MeneTools* [11], an ASP-based software checking the topological activation of reactions, and *CobraPy* [6], a python module computing metabolic flux.

All the 10% degraded tested instances finished before the timeout and returned a valid hybrid completion, *i.e.* every target reaction is topologically and stoichiometrically activated. 15 instances of the 40% degraded tested instances finished before the timeout. Each unfinished instance returned a hybrid completion, each target reaction is hybridly activated. We cannot guarantee the minimality of the returned completion. Note that results can change between two runs due to solver heuristics.

VI. Discussion

Although *Flutampl* can solve real-sized instances, the model can still be improved to decrease computation time.

In fact, it is possible to decompose the model with the *Bender decomposition* [10] or the *Dantzig-Wolfe decomposition* [8]. Bender decomposition applies the "divide and conquer" paradigm to linear programming by splitting variables in two sets and solving the problem in two times. While Dantzig-Wolfe decomposition also creates new subproblems to find a subset of useful variables. In large-scale linear problem, a great part of variables is not used for one instance, Dantzig-Wolfe decomposition tries to remove those useless variables. However, these decompositions take times to implement and do not guarantee to reduce computing time.

Another approach to reduce computation time is the *Lagrangian relaxation* [7]. This involves removing hard to satisfy constraints from the model and to add them to the objective function. The objective is to violate the constraint as less as possible. But by modifying the objective function and removing constraints, we are not sure anymore to solve the same problem. It will be necessary to add or adjust objective function's coefficients in order to get the optimal solution of the initial problem.

Conclusion

In this paper, a linear programming method to compute the recursive definition of topological activation is introduced. This definition is usually solved with logical programming method. It lets us solving the topological completion problem for metabolic networks and so the hybrid one as formulated by *C. Frioux et al* [4]. To realise it, we started by giving some initial definitions and formalism of metabolic networks, activation and completion problem. We then presented a linear model able to determine the topological activation of a reaction. This model used a flow approach to model the topological activation. We finally explained a way to optimise this model, and presented our implementation *Flutampl*. *Flutampl* is a linear programming tool computing the hybrid completion using directly biological file format (*sbml*).

For this work, we decided to implement the hybrid completion problem with linear programming. But other similar problems could be solved in linear programming with this approach as the selection of microbial consortium formulated by *A. Julien-Laferrière et al* [1]. Moreover, *Flutampl* is developed with AMPL – a proprietary software – and used CPLEX as solver. All the different existing software have their own strength and weakness, it is possible to adapt the choice of these software to improve the model performance.

Our approach of the topological activation with linear programming is a good starting point to develop new linear models using the notion of topological activation – widely used in biology. Firstly, it would be good to improve the model optimisation with the decomposition, as *Bender* [10] or *Dantzig-Wolfe* [8] decompositions. Those methods could drastically reduce computation time, and thus make the tool more convenient to use.

Acknowledgements

I would like to thank Anne Siegel for giving me this internship in the Dyliss team and guiding my work. Rumen Andonov and Sébastien François who help me understand linear programming and work with me to improve my linear model's quality. I would also like to thank all the Symbiose team[1], more particularly:

- Méziane Aite who explains me the subtlety of metabolic networks;
- Clémence Frioux who answers all my questions about *fluto*.

References

[1] Alice Julien-Laferrière et al. A combinatorial algorithm for microbial consortia synthetic design. *Nature*, 29182, 2016.

[2] Bradley, Hax and Magnanti. *Applied Mathematical Programming*, chapter 9- Integer Programming. Addison-Wesley, 1977.

[3] C. E. Miller, A. W. Tucker and R. A. Zemlin. Integer programming formulation of the traveling salesman problems. *Journal of the ACM*, 7 Issue 4:326–329, 1960.

[4] Clémence Frioux et al. Technical note hybrid metabolic network completion. *Cambridge University Press*, 2018.

[5] Iain Dunning, Joey Huchette, and Miles Lubin. Jump: A modeling language for mathematical optimization. *SIAM Review*, 59(2):295–320, 2017.

[6] Ali Ebrahim, Joshua A. Lerman, Bernhard O. Palsson, and Daniel R. Hyduke. Cobrapy: Constraints-based reconstruction and analysis for python. *BMC Systems Biology*, 7(1):74, Aug 2013.

[7] G. L. Nemhauser and L. A. Wolsey. *Integer and combinatorial optimization*, chapter Lagrangian Relaxation and Duality. Wiley-Interscience Series in Discrete Mathematics and Optimization, 1999.

[8] George B. Dantzig, Philip Wolfe. Decomposition principle for linear programs. *Operations Research*, 8 issue 1, 1960.

[9] William E. Hart, Carl D. Laird, Jean-Paul Watson, David L. Woodruff, Gabriel A. Hackebeil, Bethany L. Nicholson, and John D. Siirola. Pyomo–optimization modeling in python. *Springer Science & Business Media*, 67, 2017.

[10] J. F. Benders. Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik*, 4 issue 1, 1962.

[11] Sylvain Prigent et al. Meneco, a topology-based gap-filling tool applicable to degraded genome-wide metabolic networks. *PLOS Computational Biology*, 2017.

[12] V. Helms. *Principle of Computational Cell Biology*, chapter 6.2 Stoichiometric Matrix. WILEY-VCH, 2008.

[1] Former team which was splitted in Dyliss, GenScale and GenOuest.