Projet long : Genome Scale Metabolic Models : de l'équilibre au dynamique

Valentin Laroche – M2BI – 2019-2020

Professeur référent : Denis Mestivier

Table des matières

1. Introduction

Les GSMM (Genome-Scale Metabolic Models) sont des modèles informatiques qui décrivent de façon précise l'entièreté des associations gène-protéine-réaction dans un organisme. Ces modèles sont généralement utilisés pour décrire et étudier le métabolisme d'un organisme donné. Depuis la première apparition d'un modèle de ce type, en 1999, avec un modèle d'*Haemophilus influenzae* [1], ce domaine de la biologie des systèmes n'a fait que gagner en intérêt auprès de la communauté scientifique, qui a développé de très nombreux modèles, y compris des modèles cellulaires humains. Les applications de ces modèles sont multiples ; Parmi ces applications se retrouvent l'étude de flux métaboliques, médecine et études cliniques, développement de médicaments, ou encore des analyses du réactome [2]. En particulier, la compréhension des maladies humaines est un objectif primordial pour les GSMM, qui font déjà l'objet d'applications dans certains domaines [3, 4, 5].

Cependant, en très grande majorité, les modèles GSMM sont développés selon une méthode statique, c'est à dire que le système biologique est estimé à l'équilibre dans ses réactions. Cette assertion impose un biais non négligeable dans les données, étant donné qu'un système biologique se retrouve très rarement dans une situation d'équilibre.

Un module de simulation a donc été développé pour passer outre cette approche statique, et aller vers une approche stochastique, utilisant des règles booléennes, permettant une modélisation dynamique du système biologique. Un GSMM développé pour ce modèle est très proche d'un GSMM traditionnel, si ce n'est qu'une étape de l'étude consiste à simuler son fonctionnement sur une période de temps établie à l'avance, permettant un suivi de la cohérence de ce modèle au cours du temps. Cela pose donc de nouvelles questions et de nouveaux problèmes. En effet, une approche stochastique impose des choix à chaque étape de simulation, et deux réactions qui peuvent être biologiquement simultanées, ne seront jamais appliquées en même temps, mais seulement au sein de la même unité arbitraire de temps. Cependant, ce système de choix mène systématiquement à un échec de la simulation, et par conséquent à la « mort » du système biologique simulé, et les causes de cette échec ne sont pour le moment pas identifiées.

Il devient donc essentiel de reparamétriser ces modèles de GSMM de façon à permettre le fonctionnement du système biologique au cours du temps. Chaque règle pouvant être appliquée dans le cas où tous les réactifs sont présents, il est nécessaire d'identifier les réactions et les éléments associés qui bloquent la simulation du réseau.

Cependant, la modélisation dynamique à l'échelle du métabolisme d'un génome entier est encore un enjeu qui n'a pas encore été résolu mais qui fait l'objet de nombreuses études qui cherchent à s'en

approcher [6]. Certains modèles dynamiques ont également été développés à une échelle beaucoup plus réduite, par exemple au niveau d'interactions enzymatiques [7] montrant qu'il est possible, moyennant des études approfondies, de simuler des métabolismes de façon dynamique, à une échelle de taille de métabolisme supérieure.

L'utilisation de réseaux stochastiques évolués et optimisés permettrait la simulation, en continu, d'un système biologique complet modélisé par un GSMM, et par conséquent un suivi de l'évolution d'un système donné compte tenu du milieu dans lequel il se développe. Un tel modèle pourrait avoir de très nombreuses applications dans tous les domaines mentionnés précédemment, et résoudre des problèmes qu'un GSMM statique ne peut pas résoudre, comme la simulation de l'évolution d'une maladie, ou le potentiel pour un médicament d'impacter le réseau métabolique, ou encore le suivi des symptômes d'une maladie.

2. Matériel & Méthodes

Pour constituer la simulation et l'analyse du modèle GSMM, deux parties majeures ont été définies. Un résumé de la méthode dans son ensemble est disponible en figure 1.

La première partie concerne la simulation dynamique elle-même. Il s'agit d'une partie de développement en C qui avait déjà été réalisée avant le début de cette étude.

La seconde regroupe les deux modules d'optimisation et d'analyse des résultats de cette simulation. Ces deux modules ont été réalisés en Python 3.7 et font usage des deux modules suivants :

- Argparse, un module qui est utilisé pour la lecture des arguments lorsque l'un ou l'autre des modules est lancé.
- Textwarp un module qui permet de formater les sorties texte pour faciliter la lecture des résultats dans certains cas.

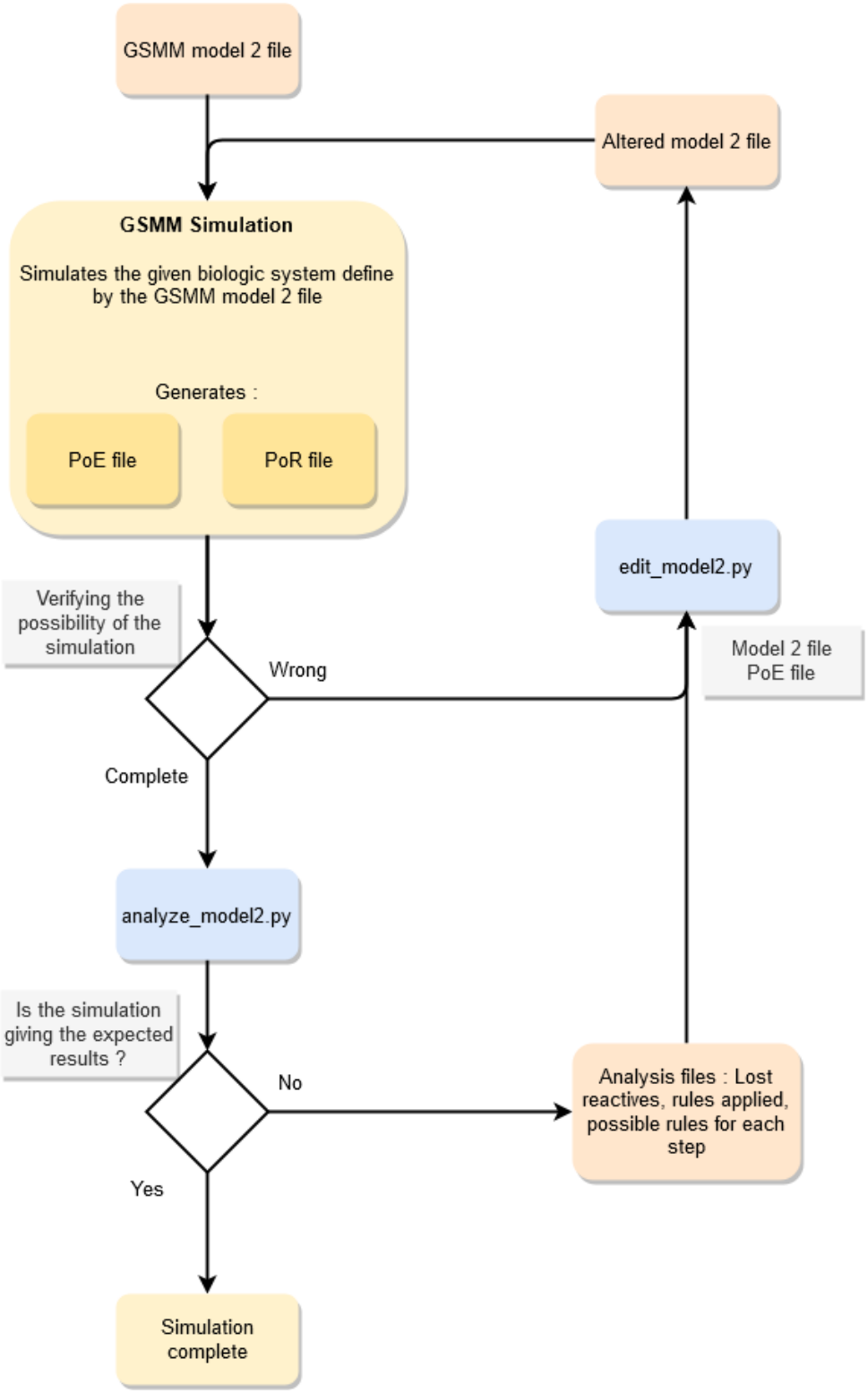

Figure 1 : Workflow de l'étude. En orange, des fichiers d'analyse ou de modèles. En bleu, les fichiers de module développés. En jaune, la simulation de GSMM.

2.1. Simulation du GSSM

L'objectif du module de simulation est de proposer une simulation d'un GSMM dans le but de prévoir le fonctionnement du système biologique associé. Un fichier « model2 » est constitué à partir

des règles qui définissent le système biologique concerné. Dans cette étude, le fichier model2 employé est un système biologique réduit constitué à partir du métabolisme d'*Escherichia Coli.* A chaque règle définie dans le fichier model2, un poids est associé, déterminant la probabilité d'une règle de se produire ; La probabilité d'activation d'une règle est proportionnelle à la grandeur de son poids. Dans chaque règle, les produits sont partiellement formés, et les réactifs sont partiellement consommés chaque fois que la règle est appelée (i.e. Il existe un pourcentage de chaque élément disponible dans le milieu, et non pas une règle binaire présent/absent). Une règle se définit comme suit :

$$Elément\ A + Elément\ B \Rightarrow Elément\ C + Elément\ D \qquad [Poids]$$

Une règle peut inclure un élément à la fois comme réactif et comme produit. (Par exemple un catalyseur, qui facilite la réaction par sa seule présence). Ce faisant, il est possible que l'élément concerné ne soit pas consommé pendant l'application de la règle. Le poids est indiqué dans le fichier à la suite de chaque règle.

Concernant le fichier model2 du système biologique d'*E. Coli*, 94 règles de ce type sont définies. Une règle « blanche » (NONE => NONE) est ajoutée pour assurer l'aboutissement de la simulation, même si aucune autre règle ne peut être utilisée, cette règle indique une « mort » du système biologique associé. Cette règle possède un poids très inférieur à tous les autres, de façon à ce qu'elle ne soit pas utilisée si d'autres règles sont possibles d'utilisation.

Une simulation est divisée en étapes de « temps », au sein desquelles un certain nombre d'évènements peuvent se produire, dépendant de leur poids (une règle de poids inférieur mettra une moins grande part de temps à s'effectuer). Chaque étape contiendra donc un ensemble d'applications de règles, dont le nombre pourra différer selon les règles sélectionnées. Il n'y a cependant pas de mesure de temps réel possible. Par défaut, une simulation se déroule pendant 20 étapes de temps équivalentes entre elles. Une simulation est considérée validée lorsque pendant ces 20 étapes, un ensemble de règles diverses (en dehors de la règle de « mort ») a été appliqué

La simulation génère nécessairement deux fichiers : Un fichier PoR « Percentage of Rules » et un fichier PoE « Percentage of Eléments ».

- Le premier, PoR, permet de suivre l'application des règles à chaque étape de la simulation. Il est possible de connaître le nombre d'évènements (règles appliquées) par étape de temps, ainsi que le pourcentage d'application de chaque règle. Par exemple, si 4 règles ont été appliquées dans la même étape de temps, une règle possédant 25% dans le fichier PoR aura été appliquée une fois. (25% du total de règles utilisées).

- Le second fichier, PoE permet quant à lui de suivre les pourcentages de chaque élément présents dans la simulation à chaque étape de temps. Un pourcentage de 100% indique que l'élément est pleinement présent, et un pourcentage égal à 0%, que l'élément a été complètement consommé.

Ces deux fichiers permettent un suivi simple des évènements qui se produisent lors d'une simulation et serviront de base aux analyses approfondies des systèmes biologiques simulés.

2.2. Elaboration du module d'optimisation : edit_model2.py

Pour optimiser les chances de réussir la simulation après un échec, un système d'édition du fichier model2 a été réalisé. Son but est de déterminer arbitrairement les règles les plus centrales à l'étude, ainsi que les composés les plus fréquemment produits ou utilisés. Ensuite, ces informations sont utilisées pour établir de nouveaux poids au système. Pour utiliser le module, un fichier model2, ainsi qu'un fichier PoE sont nécessaires.

Dans un premier temps, le module va constituer un dictionnaire (i.e. un index qui associe des « clés » à une ou plusieurs valeurs. Chaque clé fait donc référence à la, ou aux valeurs associées. Cette valeur, ou ces valeurs peuvent être des nombres, des caractères, etc…). Ce dictionnaire contient l'ensemble des règles et des poids associés à partir du fichier model2.

De plus, cette première fonction va générer un premier fichier, appelé « cytoscape_network.sif ». Ce fichier pourra être utilisé plus tard dans l'étude, pour une visualisation du réseau avec le logiciel Cytoscape. Il contient l'ensemble des interactions entre chaque élément un à un, de réactif à produit. Cet étape génèrera un dernier élément, qui contient les premières informations nécessaires à la mise en place d'un fichier d'annotation pour le réseau Cytoscape mentionné précédemment.

Ensuite, le module va utiliser le fichier PoE pour générer une liste de tous les éléments qui composent le système. Cette liste sera utilisée par la suite dans le module.

L'étape suivante consiste à constituer deux listes distinctes, l'une contenant tous les réactifs possibles dans toutes les règles et l'autre tous les produits. Un même élément est compté autant de fois qu'il apparaît dans chacune de ces listes. Ces deux listes permettent de comptabiliser le nombre total d'apparitions d'un élément à la fois dans les réactifs et dans les produits, de façon distincte. Un ratio apparition dans les réactifs / apparition dans les produits est ensuite calculé pour déterminer si un élément, à poids égaux entre toutes les règles, est plus susceptible d'être produit ou utilisé. S'il est plus susceptible d'être utilisé, alors, pour cet élément seul, il est statistiquement nécessaire que

la ou les règle(s) qui le produise(nt) soi(en)t favorisée(s) par rapport à celle(s) où il est utilisé. Un ratio est donc établi pour chacun des éléments présents dans la simulation.

Un fichier, appelé « annot_node.csv » est constitué simultanément avec cette étape. Il sera utilisé pour annoter les nœuds du réseau Cytoscape défini dans la première étape. Dans ce fichier seront écrits les noms de chaque élément, ainsi que leur nombre d'apparitions dans les réactifs, dans les produits et le ratio réactif/produit associé.

Enfin, la dernière étape d'analyse consiste à calculer l'équilibrage des poids de chaque règle, compte tenu des informations récoltées précédemment. Pour ce faire, chaque règle est analysée individuellement et un dictionnaire de valeurs est constitué ; Pour chaque règle, la moyenne des ratios réactif/produit des réactifs et des produits est calculée séparément. Ce second dictionnaire contient, pour chaque règle, deux valeurs : le ratio réactif/produit moyen des réactifs, et celui des produits. Ces deux ratios sont ensuite comparés pour établir de nouveaux poids selon plusieurs cas possibles, comme le montre la figure X. Dans le cas où aucune de ces conditions n'est remplie, le poids est maintenu à sa valeur initiale.

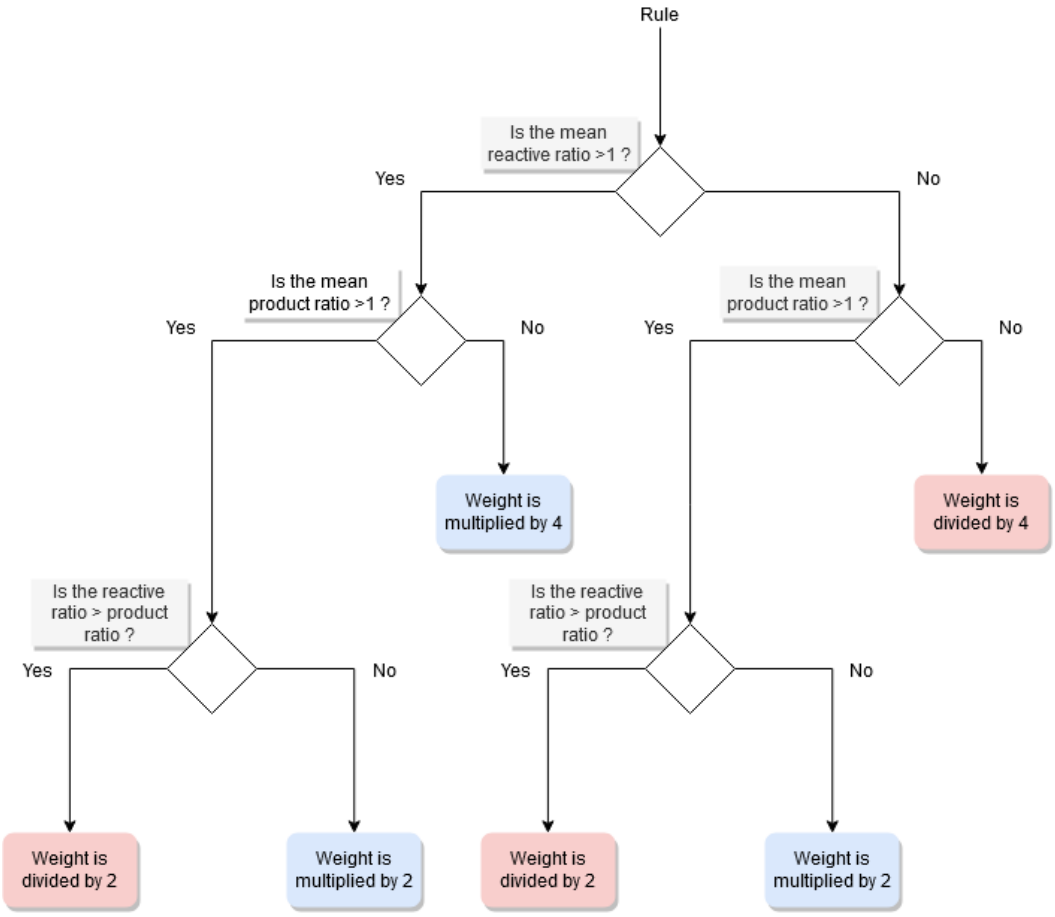

Figure 2 : Arbre de décision de modification des poids en fonction des ratios moyens réactif/produit de chaque règle. En bleu est représentée une augmentation, en rouge une diminution par rapport au poids initial.

Un nouveau fichier model 2 est donc généré, suivant les mêmes règles que le fichier d'origine, mais avec des poids altérés, de façon à orienter la simulation de façon à obtenir plus fréquemment ou non les règles, de façon à optimiser la conservation du plus grand nombre possible d'éléments. Ce fichier model 2 est donc utilisé pour une réitérer une simulation, afin de poursuivre les analyses, celui-ci étant statistiquement plus fiable qu'un modèle accordant des poids équivalents à toutes les règles. Dans le même temps, un fichier « annot_edge.csv » permettant d'annoter les ponts entre deux nœuds du réseau Cytoscape est généré. Celui-ci contient, pour chaque interaction entre deux éléments, l'ancien et le nouveau poids associé à chaque règle qui définit cette interaction.

Une comparaison sera faite entre un fichier model 2 initial, et un fichier model 2 modifié par ce module d'optimisation vis-à-vis de la qualité et de la réussite de la simulation.

2.3. Elaboration du module d'analyse : analyze_model2.py

Dans le cas où cet équilibrage des poids n'est pas suffisant, d'autres analyses sont réalisées dans un second module qui vise à cibler les points de fragilité du modèle. Une liste, non exhaustive, des possibles fragilités pourrait être :

- Un élément est entièrement consommé et n'est plus, ou ne peut plus être produit
- Une règle ne peut plus être appliquée du fait d'un élément manquant
- Un élément manque à l'origine du système et ne peut pas être produit
- Un poids trop peu équilibré a été accordé à une règle qui prend donc trop, ou trop peu d'importance.

Pour pallier ces problèmes, il est nécessaire d'identifier tous les points qui ont bloqué la simulation. L'application de ce module nécessite cette fois le fichier model2, et les deux fichiers de résultats PoE et PoR. Enfin, le module nécessite également le fichier d'annotation des nœuds, pour récupérer des information sur les éléments du modèle.

Dans un premier temps, comme pour le module d'optimisation, un dictionnaire est créé pour contenir les règles et les poids correspondants.

Ensuite, le module utilise le fichier PoR pour générer un dictionnaire qui repère d'une part, pour chaque étape de temps, toutes les règles qui ont été employées au moins une fois, et d'autre part totalise le nombre de fois où chaque règle a été utilisée, encore une fois à chaque étape de temps. Ces informations sont répertoriées dans un fichier « rules_applied.txt ».

Deux autres fichiers, « possible_rules.txt » et « lost_reactives.txt » sont générés à partir du dictionnaire de règles et du fichier PoE. Le module repère, à chaque étape, tous les éléments

présents dans le milieu biologique simulé. Il confronte ensuite ces éléments présents au dictionnaire contenant toutes les règles de la simulation, de façon à déterminer l'ensemble des règles qui pourraient être appliquées au début de chaque étape de temps. D'autre part, cette partie du module permet également de repérer les éléments manquants à chaque étape de la simulation, qui bloquent donc l'usage de toutes les règles dont ils sont les réactifs.

Enfin, une dernière partie du module permet de faire un résumé de la dernière étape de la simulation à partir des éléments manquants et du fichier d'annotation des nœuds pour repérer, à la fin de la simulation. L'intérêt particulier de cette fonctionnalité et de repérer, dans une simulation qui n'a pas abouti à un résultat cohérent, tous les éléments qui ont participé à placer la simulation dans un cas qui bloque son fonctionnement logique et d'observer quels sont leurs ratios réactif/produit. Il devient donc possible de remonter aux règles, en particulier pour les éléments entièrement consommés alors que leur ratio réactif/produit était très bas et incitait donc le modèle à réduire le poids des règles qui les utilisaient.

L'utilisation de ces fichiers d'analyse permet donc d'adapter les modules d'optimisation, et de comprendre les fragilités qui peuvent être induites par la sélection de poids arbitraire, qui pourrait desservir certains systèmes très complexes.

2.4. Visualisation sur Cytoscape

Afin d'assister l'étude d'une simulation, en particulier lorsqu'il s'agit de réseaux de règles très complexes, il devient primordial d'obtenir une visualisation de toutes les interactions impliquées dans le réseau. Pour obtenir une telle visualisation, le logiciel Cytoscape, permettant la visualisation simple et personnalisée de réseaux biologiques et d'interactions moléculaires va être utilisé.

Pour ce faire, trois fichiers distincts ont été générés lors de l'application du module d'optimisation. Le fichier principal permettant la génération du réseau (i.e. « cytoscape_network.sif ») ainsi que deux fichiers d'annotation (i.e. « annot_edge.csv » et « annot_node.csv ») destinés à, respectivement, annoter les ponts et les nœuds dans le réseau. Ces fichiers permettent au logiciel Cytoscape de constituer un réseau d'interactions qui représente l'intégralité des règles du modèle. Chaque « nœud » correspond à un élément du modèle GSMM, tandis que chaque « pont » correspond à une interaction qu'un élément donné possède avec un autre. Ces ponts permettent de voir quel élément est en interaction, comme réactif, ou produit, d'un autre élément sans mention précise de la règle à laquelle ces deux éléments sont associés.

L’utilisation d’un tel réseau permet de quantifier, de façon visuelle, les interactions les plus importantes, du fait du poids de la règle à laquelle elles sont associées, ainsi que les ratios réactif/produit de chaque élément.

3. Résultats

3.1. Simulation du GSSM

Les premières simulations du modèle d'*E. Coli*, avant toute optimisation, ont abouti à une conclusion simple : Une grande majorité des règles ne pouvaient pas être appliquées du fait du manque d'environ la moitié des composés à l'état initial. De fait, la quasi-totalité des règles étaient rendues impossibles et le modèle basculait par défaut sur la règle de « mort » NONE => NONE après seulement une ou deux étapes de simulation. Pour rappel, toutes les règles pouvaient se réaliser de façon équivalente, mis à part la règle NONE => NONE.

L'échec de ces simulations ne pouvait être expliqué, à première vue, que par les éléments manquants, ou les règles qui n'étaient pas sélectionnées, ce qui représente des solutions très floues. Par conséquent, pour obtenir des résultats plus visuels dans les causes d'échec des simulations. Le choix arbitraire de laisser tous les éléments existants dans l'ensemble de règles présents au début de la simulation a donc été fait, pour observer comment se déroulait la simulation lorsque toutes les règles étaient envisageables au début de la simulation.

Cette modification n'a cependant pas eu d'impact sur la réussite de la simulation, qui se retrouvait dans le même cas évoqué précédemment dans les mêmes délais. Il devient donc nécessaire d'identifier les causes précises des échecs, qui résident sûrement dans la pondération des règles et l'utilisation des éléments.

3.2. Module d'optimisation : edit_model2.py

La première solution possible pour compléter correctement les simulations est donc d'accorder des poids variables pour inciter la simulation à utiliser des règles essentielles au bon fonctionnement du système biologique plus fréquemment, ou encore d'éviter d'employer trop celles qui consomment un élément qui ne peut pas se régénérer au cours de la simulation.

L'utilisation du module d'édition a permis de modifier une grande majorité des règles du modèle, augmentant ou réduisant les poids sur une échelle allant de 0.25 à 4 selon une moyenne de ratios réactif/produit.

Cette solution, relativement simple, n'a pas permis à la simulation de s'achever dans des circonstances satisfaisantes. Cependant, le nombre d'étapes de temps que parcourt la simulation avant d'échouer (i.e. appliquer uniquement la règle de « mort ») se situe entre 5 et 7 étapes, selon la simulation, ce qui pointe tout de même une solution potentielle, bien que non suffisante.

D’autre part, certaines simulations ont présenté des résultats très particuliers, pouvant s’apparenter à des minima locaux. En effet, dans certaines conditions précises, le fichier model2 modifié, une fois soumis à la simulation, a, en de rares occasions réussi à trouver des relations d’équilibre en utilisant 2 ou 3 règles auto-suffisantes dans le système, ignorant les autres. Il s’agit d’un point d’intérêt notable, malgré le fait que ces minima locaux soient inexploitables d’un point de vue biologique.

3.3. Module d’analyse : analyze_model.py

En utilisant le fichier model 2 modifié, accompagné des fichiers PoE et PoR générés par la simulation il est possible d’identifier, pour chaque simulation, les éléments manquants et les règles possibles et appliquées.

Après une vingtaine de simulations sur le même fichier model 2, aucun élément ne semble véritablement récurrent dans les causes d’échec de la simulation. Il est possible d’en déduire que le modèle lui-même n’est pas valide pour réussir la simulation. En effet, de nombreux cheminements conduisent, plus ou moins rapidement, à la même conclusion : Un système fonctionnant pendant toute la durée de la simulation, sans atteindre de minimum local, n’est pas possible étant donné la configuration actuelle du fichier model 2. Cependant, plusieurs éléments ont pu être mis en évidence comme étant uniquement des réactifs, et n’étant jamais produits, alors qu’ils sont centraux au niveau du système biologique.

D’autre part, il est intéressant de noter que certains éléments sont très vite consommés par le système, et si aucune règle ne vient compenser cette perte, une grande partie du système peut rapidement se retrouver bloquée. Pour des raisons techniques, des modifications internes au module de simulation n’ont pas pu être étudiées ni appliquées. Mais il s’agit d’un point d’étude qu’il faut adresser dans le futur afin de garder le maximum de liberté dans les modifications qui peuvent être apportées au module de simulation dans son ensemble.

3.4. Visualisation sur Cytoscape

La visualisation du modèle sur le logiciel Cytoscape permet une très bonne compréhension du système correspondant à l’étude. En effet, comme le montre la figure X, il est possible de remarquer intuitivement les points névralgiques du système biologique d’*E. Coli*. On remarque également qu’une majorité des éléments n’interagissent qu’une fois avec le système, et se régénèrent dans le même temps, montrant que le fonctionnement du système ne tient qu’à plusieurs points centraux qui doivent nécessairement être possibles d’utilisation.

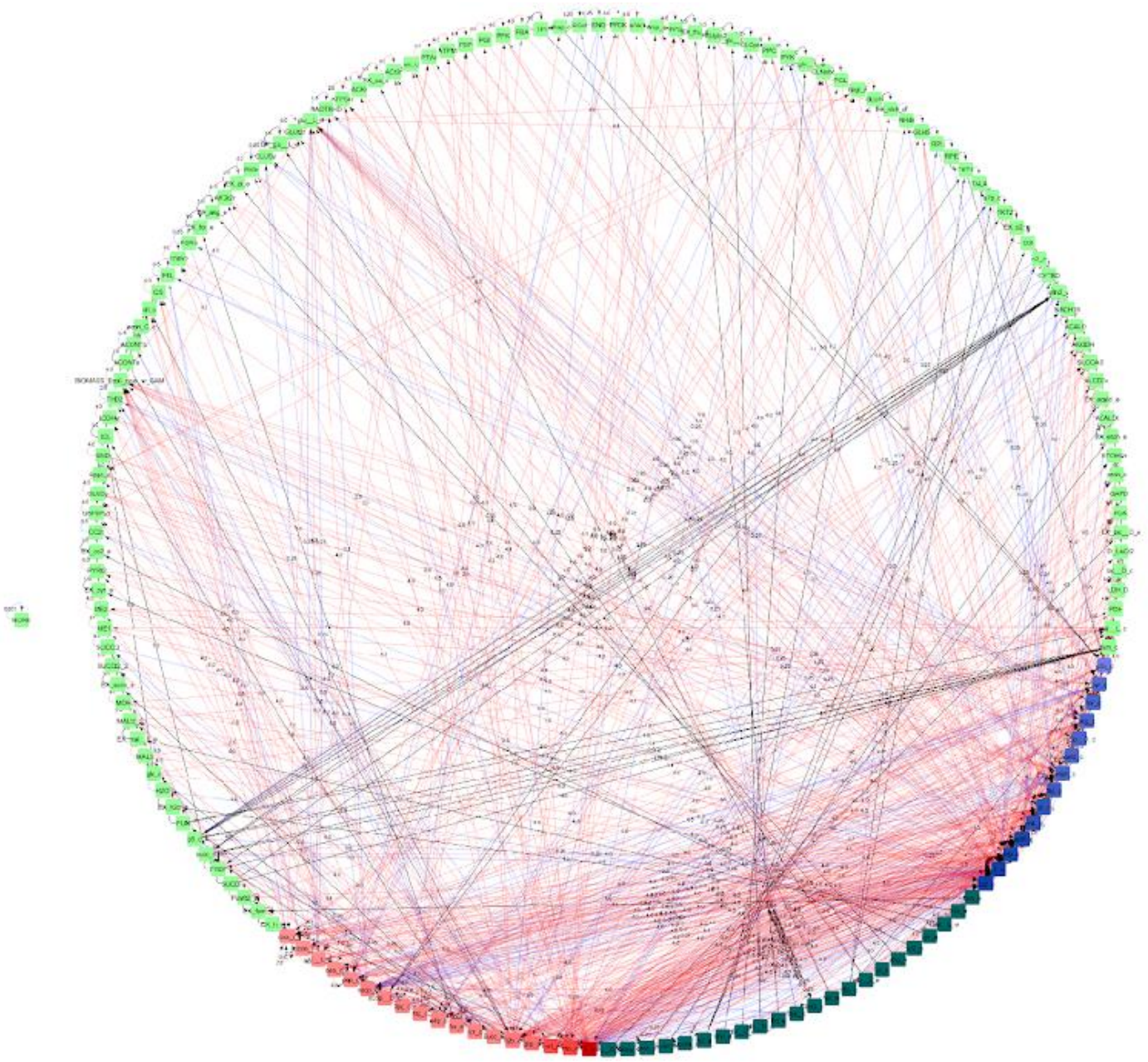

Figure 3 : Représentation du système biologique d'intérêt avec le Logiciel Cytoscape. Pour les nœuds : En vert, les éléments du modèle avec un ratio réactif/produit égal à 1. En nuances de rouge, les éléments avec un ratio supérieur à 1 (plus utilisés que produits à poids égaux), un rouge plus intense représentant un ratio plus élevé. En nuances de bleu, un ratio inférieur à 1 (plus produits qu'utilisés à poids égaux), un bleu plus intense représentant un ratio plus faible. En bleu-vert, un ratio égal à 0, indiquant soit un défaut de présence de l'élément dans les réactifs ou dans les produits. Pour les ponts : En noir, les interactions issues d'une règle avec un poids égal à 1. En rouge, les interactions issues d'une règle avec un poids supérieur à un. En bleu, les interactions issues d'une règle avec un poids inférieur à 1. L'élément isolé sur la gauche de la figure principale est la règle de « mort » NONE => NONE.

4. Conclusion & perspectives

Cette étude a pu montrer qu'il était possible de retirer beaucoup de pistes d'amélioration depuis le module de simulation. D'une part, il est possible de modifier les poids de façon plus cohérente en se basant uniquement sur la quantification de chaque élément dans les modèles proposés. D'autre part, il est possible, de clairement identifier les éléments et règles essentielles du modèle à l'aide d'une visualisation du réseau par le logiciel Cytoscape. Enfin, un ensemble de fichiers d'analyse peut être généré pour chaque modèle, afin d'identifier précisément les éléments et les règles qui ont bloqué la simulation du GSMM.

Il reste cependant à exploiter ces résultats obtenus pour altérer le fichier model 2 par la suite. Intégrer ces informations nécessite une connaissance technique plus avancée du module de simulation et des procédés biologiques impliqués pour proposer des réponses cohérentes aux causes de « mort » prématurée du GSMM. Altérer à nouveau les poids semble également une idée de choix du fait d'une amélioration notable du temps de survie du système, ainsi que des minima locaux trouvés lors de certaines simulations.

Analyser d'autres systèmes plus étendus représentant le métabolisme d'*E. Coli* pourrait également permettre de trouver d'autres règles supplémentaires pour compenser les divers cas « finaux » qui consomment un élément, alors qu'il ne peut être produit par aucune autre règle. La disparition de ces éléments entraîne le blocage de toutes les règles auxquelles ils sont associés, réduisant la quantité de règles applicables aux étapes de simulation suivant leur disparition.

Il serait également possible de définir un troisième module d'analyse permettant de corréler les poids fixés par le module d'optimisation avec les données d'analyse développées à la suite de la simulation des réactions pour réitérer la simulation avec des poids plus précisément attribués. Lorsqu'une compréhension plus profonde du fonctionnement du système aura été obtenue, il sera possible d'utiliser l'ensemble de ces modules pour simuler dynamiquement des systèmes métaboliques, de taille plus élevée, en approchant une plus forte proximité biologique que celle proposée par les modèles statiques existants.

5. Références

Il est possible de trouver les deux modules Python ainsi que ce rapport, et des fichiers d'exemples pour utiliser les deux modules développés dans cette étude à l'adresse suivante : https://github.com/ValLaroche/ProjetLong

[1] Edwards, Jeremy S., et Bernhard O. Palsson. « Systems Properties of the Haemophilus InfluenzaeRd Metabolic Genotype ». *Journal of Biological Chemistry* 274, nº 25 (18 juin 1999): 17410-16. https://doi.org/10.1074/jbc.274.25.17410.

[2] Gu, Changdai, Gi Bae Kim, Won Jun Kim, Hyun Uk Kim, et Sang Yup Lee. « Current status and applications of genome-scale metabolic models ». *Genome Biology* 20, nº 1 (13 juin 2019): 121. https://doi.org/10.1186/s13059-019-1730-3.

[3] Guebila, Marouen Ben, et Ines Thiele. « Predicting Gastrointestinal Drug Effects Using Contextualized Metabolic Models ». *PLOS Computational Biology* 15, nº 6 (26 juin 2019): e1007100. https://doi.org/10.1371/journal.pcbi.1007100.

[4] Heinken, Almut, Dmitry A. Ravcheev, Federico Baldini, Laurent Heirendt, Ronan M. T. Fleming, et Ines Thiele. « Systematic assessment of secondary bile acid metabolism in gut microbes reveals distinct metabolic capabilities in inflammatory bowel disease ». *Microbiome* 7, nº 1 (15 mai 2019): 75. https://doi.org/10.1186/s40168-019-0689-3.

[5] Noronha, Alberto, Jennifer Modamio, Yohan Jarosz, Elisabeth Guerard, Nicolas Sompairac, German Preciat, Anna Dröfn Daníelsdóttir, et al. « The Virtual Metabolic Human Database: Integrating Human and Gut Microbiome Metabolism with Nutrition and Disease ». *Nucleic Acids Research* 47, nº D1 (8 janvier 2019): D614-24. https://doi.org/10.1093/nar/gky992.

[6] Gilbert, David, Monika Heiner, Yasoda Jayaweera, et Christian Rohr. « Towards Dynamic Genome-Scale Models ». *Briefings in Bioinformatics* 20, nº 4 (19 juillet 2019): 1167-80. https://doi.org/10.1093/bib/bbx096.

[7] Tonn, Mona K., Philipp Thomas, Mauricio Barahona, et Diego A. Oyarzún. « Stochastic Modelling Reveals Mechanisms of Metabolic Heterogeneity ». *Communications Biology* 2, nº 1 (21 mars 2019): 1-9. https://doi.org/10.1038/s42003-019-0347-0.