

Rapport de Projet

Comparaison de molécules : Similarité des graphes de cycles

DEMANGE Noé - 21905012
DOCHERTY Ronan - 21800639
FERNET Anne - 21915170
NESSI Victor - 22302737



M2-AMIS | Module Projet
UFR des Sciences | Université de Versailles Saint-Quentin / Paris-Saclay
France
Année universitaire 2023 - 2024

Table des matières

1	Introduction	3
2	Modèle et Algorithmes	3
2.1	Graphe de Cycles	3
2.1.1	Modélisation	4
2.1.2	Algorithme de McKay	6
2.1.3	Algorithme de Horton	6
2.2	Similarité des Graphes de Cycles	7
2.2.1	Distance d'édition	7
2.2.1.1	Entre les graphes de cycles	7
2.2.1.2	Entre les chaînes de caractères représentant les atomes	8
2.2.2	Sous-graphe commun maximum	8
3	Matériels et Méthodes	11
3.1	Récupération et extraction des données	11
3.2	Structures de données	12
3.3	Base de cycles minimale canonique d'un graphe	13
3.4	Transformation en graphe de cycles	14
3.5	Comparaison des graphes de cycles moléculaires	15
3.5.1	MCIS	15
3.5.2	Levenshtein	16
3.6	Analyse des Similarités	18
3.6.1	Calcul des Distances	18
3.6.2	Classification Hiérarchique	18
3.6.3	Heatmap	18
4	Résultats et Analyse	18
4.1	Examen de la sélection de la métrique de distance	19
4.2	Analyse des approches de similarité : MCIS, distance de Levenshtein, et combinaison des deux	20
4.3	Analyse des classes d'équivalences	23
5	Perspectives	26
6	Conclusion	27
	Annexe A Clusters	29

1 Introduction

Dans le cadre du Master 2 *Algorithmique et Modélisation à l'Interface des Sciences*, notre projet se focalise sur la comparaison de molécules en utilisant la modélisation par graphe de cycles. Cette approche s'inscrit parfaitement dans les thématiques de notre programme de master, conjuguant algorithmique et modélisation appliquée à la chimie, constituant ainsi un objet d'étude de la chemo-informatique.

La chemo-informatique, domaine bi-disciplinaire, applique les principes de l'informatique aux problèmes de la chimie. Son objectif est de fournir des outils et des méthodes pour l'analyse et le traitement des données chimiques, particulièrement cruciaux dans le contexte actuel où la recherche génère une quantité colossale de données. Parmi les nombreux défis de la chemo-informatique, la comparaison de molécules revêt une importance particulière. Des méthodes efficaces sont nécessaires pour cette comparaison, en vue de son application aux bases de données de molécules, afin d'aider les chimistes à découvrir des molécules similaires à celles qu'ils étudient.

Traditionnellement, la comparaison de molécules implique l'analyse de leurs graphes moléculaires, où les atomes sont des sommets et les liaisons sont des arêtes. Cependant, les méthodes basées sur l'isomorphisme et d'autres problèmes de la classe NP sont souvent inefficaces pour des analyses rapides des molécules. Afin de remédier à ces limitations, nous explorons une approche alternative en nous concentrant sur les graphes de cycles. Ces derniers offrent une représentation structurelle des molécules, où chaque sommet représente un cycle de la molécule et les arêtes décrivent l'interconnexion de ces cycles. Cette représentation offre une vue d'ensemble de la molécule à une échelle plus large, permettant une comparaison plus efficace.

Dans ce projet, nos objectifs sont multiples. Nous cherchons à comprendre les méthodes existantes de modélisation de molécules et d'étude de la similarité de graphes, ainsi qu'à les appliquer aux molécules de la base de données ChEBI [Hastings et al., 2016]. En outre, nous explorons de nouvelles idées et l'application d'autres algorithmes pour améliorer la comparaison de molécules.

Pour réaliser ces objectifs, nous détaillerons dans la section 2 Modèle et Algorithmes la base de données ChEBI ainsi que les différents algorithmes que nous utiliserons, tels que ceux de McKay et Horton, le MCS (Maximum Common Subgraph), etc. Dans la section 3 Matériels et Méthodes, nous présenterons notre propre implémentation. Ensuite, nous analyserons les résultats obtenus dans la section 4 Résultats et Analyse. Enfin, nous envisagerons des pistes d'amélioration dans la section 5 Perspectives, avant de conclure notre projet.

2 Modèle et Algorithmes

Dans cette partie, nous allons présenter la modélisation adoptée pour représenter les graphes de cycles des molécules ainsi que les algorithmes utilisés afin de modéliser les graphes de cycles et de les comparer.

Dans la suite du rapport, nous considérons toujours des graphes non-orientés définis par $G = (V, E)$ avec V l'ensemble des sommets et E l'ensemble des arêtes.

2.1 Graphe de Cycles

Les graphes de cycles permettent de capturer l'aspect structurel d'une molécule et peuvent être comparés pour déterminer un coefficient de similarité entre les molécules. Cette représentation plus compacte que celle des graphes moléculaires a également pour intérêt de limiter le temps de calcul sur les comparaisons, car les algorithmes utilisés ont pour certains une complexité exponentielle.

Afin de déterminer le niveau de similitude de deux molécules par comparaison de leur graphe de cycles, il faut que la représentation de ces graphes soit unique. En effet, si une molécule possédait plusieurs représentations par graphe de cycles, il pourrait résulter de leur comparaison que la molécule n'est pas strictement identique à elle-même. Cette condition sur le graphe des cycles demande de trouver une représentation dite canonique des graphes de cycles. En outre, deux molécules possédant une structure très similaire doivent avoir un niveau de similitude élevé en comparant leur graphe de cycles.

Ces conditions posent le problème du choix des cycles qui composent le graphe de cycles d'une molécule, ainsi que celui de la représentation de leur interconnexion. Nous présentons ci-dessous une modélisation des graphes de cycles inspirée de [Nouleho ilemo, 2020], qui répond à ces critères.

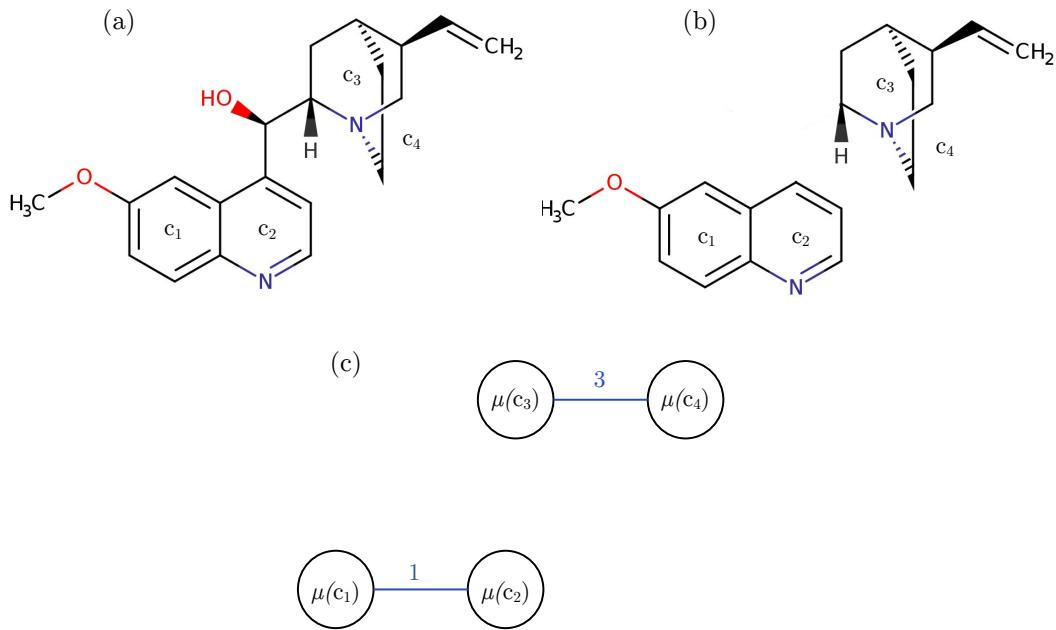


FIGURE 1 – (a) Graphe moléculaire de la quinine, (b) Graphe moléculaire d'une autre molécule, (c) Graphe de cycles obtenu pour les deux molécules si l'on ne tient pas compte des arêtes de type 2.

2.1.1 Modélisation

La modélisation en graphe de cycles d'une molécule doit refléter au mieux sa structure. Elle ne doit donc pas perdre les informations essentielles telles que la taille des cycles et la "distance" entre ces cycles dans la molécule.

Les sommets du graphe, qui correspondent aux cycles, peuvent donc être étiquetés par la taille, en nombre d'atomes, de chaque cycle. Pour représenter leur interconnexion, il est possible de définir deux types d'arêtes : un type qui va modéliser la connexion entre les cycles qui partagent des sommets en commun, et un type qui représente la connexion entre les cycles reliés par une chaîne d'atomes n'appartenant à aucun cycle. Ce second type est important pour distinguer les graphes de cycles des molécules connexes de celles qui ne le sont pas. Ce cas de figure est illustré en figure 1. En figure 2 est présenté le graphe de cycles de la quinine donné par cette modélisation. Les sommets sont étiquetés par la taille des cycles, et les arêtes sont étiquetées par le nombre d'arêtes en commun entre les cycles (type 1), ou le nombre d'arêtes qui les séparent (type 2).

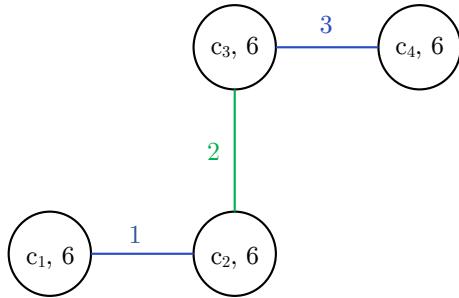


FIGURE 2 – Graphe de cycles de la quinine tel que défini par notre modélisation. Les arêtes en bleu représentent des arêtes de type 1, et celle en vert une arête de type 2.

Il reste maintenant à adresser le choix des cycles qui feront partie du graphe de cycles. Dans l'exemple de la quinine, le choix a été de garder tous les plus petits cycles qui ne sont pas contenus dans des cycles plus grands. Il s'agit du choix intuitif qui semble le plus cohérent avec la nature chimique des objets que nous manipulons. En effet, comme présenté en figure 3b, nous aurions pu conserver le cycle formé par la somme disjointe des cycles c_1 et c_2 , et le substituer à c_1 . Mais ce cycle ne semble pas aussi pertinent.

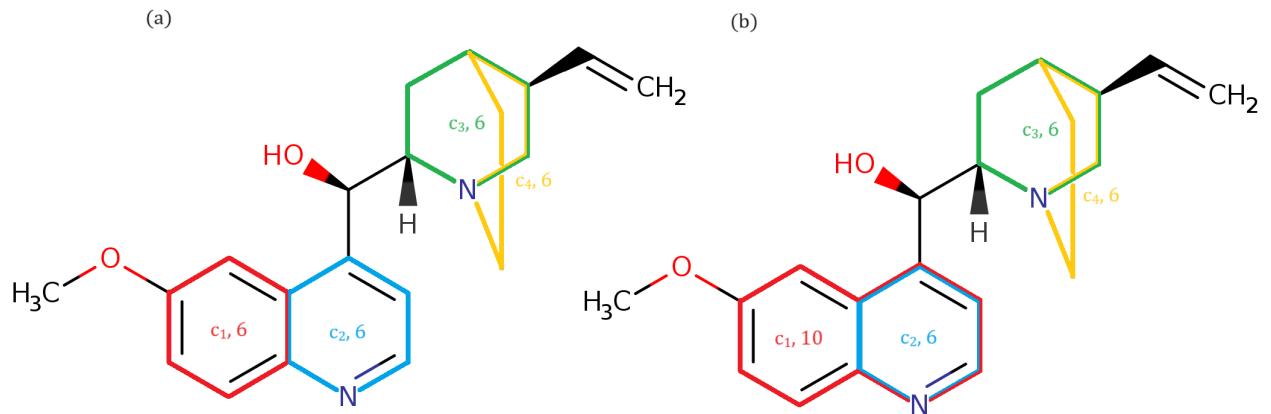


FIGURE 3 – Choix des cycles dans la molécule de quinine. Les cycles sélectionnés pour faire partis du graphe de cycles sont mis en évidence par des couleurs : c_1 (rouge), c_2 (bleu), c_3 (vert) et c_4 (jaune). (a) Cycles de la modélisation présentée en figure 2, (b) Autres cycles pouvant être utilisés pour la modélisation. En particulier, c_1 est obtenu en faisant la somme disjointe des cycles rouge et bleu en (a), et compte maintenant 10 atomes.

On peut remarquer que chacun des cycles présentés en 3a et en 3b ne peut pas être obtenu par combinaison de deux autres cycles de l'ensemble auquel il appartient. Les cycles sont alors dits indépendants. Ces deux ensembles forment ce que l'on appelle une base de cycles.

Dans un graphe, une base de cycles définie un ensemble de cycles indépendants à partir desquels tous les cycles du graphe peuvent être générés par somme disjointe. Toutes les bases de cycles d'un graphe contiennent le même nombre de cycles, qui est de $m - n + p$, avec m le nombre d'arêtes, n le nombre de sommets et p le nombre de composantes connexes. De plus, la base de cycle est dite minimale quand la somme des tailles de cycles qui composent la base est minimale.

Dans l'exemple de la figure 3, la base de cycles (a) est minimale. Une base de cycles minimale est donc un bon candidat pour construire le graphe de cycles d'une molécule. Cependant, un même graphe peut posséder plusieurs bases de cycles minimales. C'est le cas du graphe moléculaire de la quinine. Il

existe trois bases de cycles minimales, comme illustré en figure 4. Et les graphes de cycles associés à ces bases ne sont pas identiques.

Pour utiliser une base de cycles minimale comme représentant canonique des cycles d'un graphe moléculaire, il faudrait être capable de discriminer une base en particulier. Or, dans l'algorithme de génération d'une base de cycles minimale que nous présentons en sous-section 2.1.3, le choix de la base dépend de l'ordre dans lequel les sommets du graphe sont parcourus. Afin de s'assurer de l'unicité de la base de cycles, il est possible de faire une numérotation canonique des sommets des graphes moléculaires (sous-section 2.1.2).

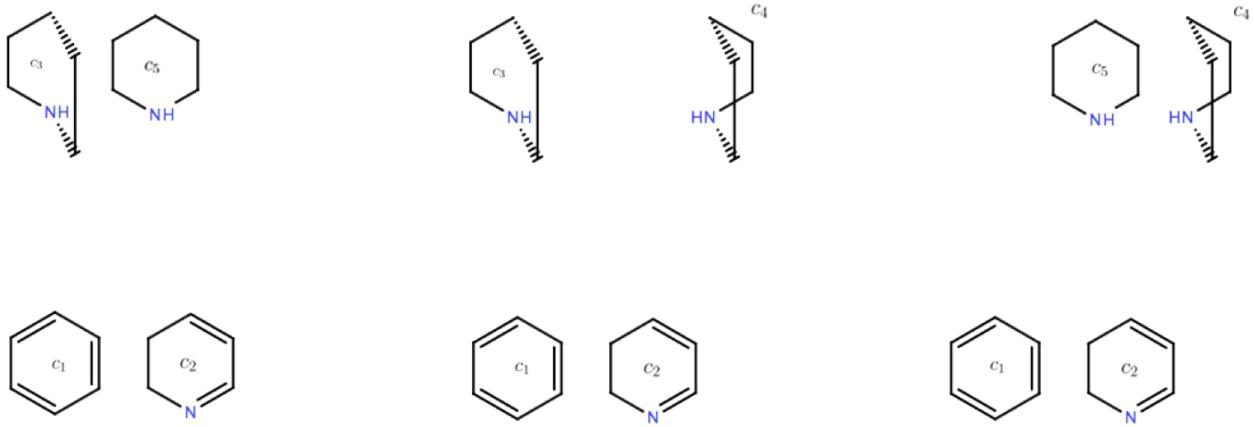


FIGURE 4 – Les trois bases de cycles minimales du graphe moléculaire de la quinine

2.1.2 Algorithme de McKay

L'algorithme de Brendan McKay sert à résoudre le problème d'isomorphisme de graphes. Deux graphes sont isomorphes s'il existe une application bijective entre les sommets des deux graphes qui préserve la structure des graphes. Il n'existe aujourd'hui pas d'algorithme polynomial pour résoudre ce problème. Cependant, l'algorithme de McKay s'effectue en temps polynomial pour les graphes de degré borné, comme c'est le cas pour les graphes moléculaires [Hartke and Radcliffe, 2013].

Cet algorithme consiste à trouver la numérotation canonique des sommets d'un graphe. En effet, si deux graphes sont isomorphes, leurs sommets auront la même numérotation canonique. La numérotation canonique des sommets d'un graphe est associée au plus petit identifiant binaire que l'on peut obtenir en faisant le XOR des lignes de la matrice d'adjacence du graphe. Brièvement, cet algorithme cherche à partitionner les sommets en fonction de leur degré, jusqu'à l'obtention d'une partition discrète, où chaque bloc ne contient qu'un sommet. Cette partition finale donne le tableau des permutations à effectuer sur les sommets du graphe pour qu'il soit canonique.

McKay a implémenté cet algorithme dans la bibliothèque *nauty*, écrite en C [McKay and Piperno, 2014].

2.1.3 Algorithme de Horton

L'algorithme de Horton a été le premier à fournir une méthode pour trouver une base de cycles minimale dans un graphe en un temps polynomial [Horton, 1987]. Cet algorithme se base sur le théorème suivant :

Soit x un sommet quelconque d'un cycle c appartenant à une base de cycles minimale. Il existe une arête

$[y, z]$ dans c , telle que c est constitué d'un plus court chemin entre x et y , d'un plus court chemin entre x et z et de l'arête $[y, z]$.

Dans le cas des graphes moléculaires, une chaîne constitue un plus court chemin entre deux sommets si elle possède un nombre minimale d'arêtes.

En utilisant cette propriété, l'algorithme de Horton cherche un plus court chemin entre toutes les paires de sommets. Puis, pour chaque sommet x et chaque arête $[y, z]$ du graphe, l'algorithme vérifie si les plus courtes chaînes entre x et y et entre x et z sont arête-disjointes. Auquel cas, le cycle formé par les plus courtes chaînes et l'arête $[z, y]$ est conservé. Tous les cycles trouvés sont ordonnés par ordre de taille croissante, et un algorithme glouton est appliqué afin de récupérer une base de cycles minimale à partir de ces cycles. Une élimination de Gauss binaire est classiquement utilisée pour extraire la base de cycles minimale de la liste ordonnée des cycles.

Comme expliqué dans la section 2.1.1, l'algorithme de Horton nécessite de parcourir les sommets dans un certain ordre. En ordonnant les sommets de manière canonique avec l'algorithme de McKay, l'algorithme de Horton devrait ensuite retourner la même base de cycles pour deux graphes moléculaires possédant une numérotation canonique similaire. Afin d'assurer la canonicité de la base de cycles minimale, il est pertinent de réduire les graphes moléculaires aux atomes constituant les cycles et les chaînes reliant ces cycles. Autrement dit, de ne pas considérer les atomes qui n'interviennent pas dans la structure cyclique du graphe. Ainsi, seuls les atomes "pertinents" sont numérotés, et deux graphes moléculaires réduits identiques ont également des graphes de cycles identiques.

Une fois une base de cycles minimale obtenue, les cycles peuvent être transformés en graphe de cycles en utilisant la modélisation présentée plus haut.

2.2 Similarité des Graphes de Cycles

Après obtention du graphe de cycles de toutes les molécules, notre objectif principal est de comparer ces graphes entre eux. Pour cela, nous avons exploré diverses méthodes de comparaison de graphes. Le choix des méthodes de comparaison dépend de plusieurs facteurs, notamment du type de graphe, par exemple si le graphe est étiqueté, de sa taille, ainsi que de l'aspect de la structure que nous souhaitons étudier.

L'isomorphisme de graphe est l'une des méthodes qui peut être employée pour comparer deux graphes. Cependant, cette méthode présente une limitation majeure : elle fournit une réponse binaire, indiquant simplement si les deux graphes sont identiques ou différents. Notre choix s'est donc porté sur d'autres approches pour évaluer la similarité entre les graphes.

Parmi ces approches, nous nous sommes intéressés aux distances d'édition appliquées aux graphes (GED) ainsi qu'aux méthodes de sous-graphes communs maximum (MCS). Les distances d'édition permettent de quantifier les différences structurelles entre les graphes, tandis que les méthodes de sous-graphes communs maximum identifient les sous-structures partagées entre deux graphes, offrant ainsi une mesure de similarité plus fine et plus nuancée.

2.2.1 Distance d'édition

2.2.1.1 Entre les graphes de cycles

Ces méthodes cherchent à quantifier les différences structurelles entre les graphes, s'inspirant des méthodes de distances d'édition sur le texte. Toutefois, choisir la méthode appropriée peut être un

défi, car le problème de calcul de la distance d'édition entre deux graphes est NP-difficile. Cela signifie qu'il est difficile de trouver une solution exacte dans un temps raisonnable, surtout pour les graphes de grande taille. En conséquence, de nombreuses heuristiques ont été développées pour trouver des solutions approchées efficaces [Blumenthal et al., 2020].

De plus, pour appliquer ces méthodes, il est nécessaire de définir des coûts pour les différentes opérations d'édition, telles que l'insertion, la suppression ou la modification d'arêtes ou de sommets. Déterminer ces coûts de manière appropriée est complexe, car ils influencent directement les résultats de la comparaison. Une mauvaise attribution des coûts peut conduire à des estimations de similarité erronées.

Ces deux problèmes, le choix d'une heuristique adaptée et des coûts appropriés, nous ont freinés dans le choix de cette méthode. Nous avons ainsi préféré nous orienter vers une méthode où le nombre de paramètres à définir était limité.

2.2.1.2 Entre les chaînes de caractères représentant les atomes

La distance de Levenshtein est une distance d'édition qui s'applique sur des chaînes de caractères [Levenshtein, 1966]. Cette distance d'édition a retenu notre intérêt, car elle peut être utilisée pour comparer les chaînes de caractères décrivant le type des atomes qui constituent les molécules.

Cette distance utilise une formule de programmation dynamique pour déterminer le nombre d'opérations (insertion, suppression, substitution) nécessaires pour passer d'une chaîne de caractères à une autre.

Combinée à une mesure de similarité entre les graphes de cycles, elle offre la possibilité de tenir compte de la composition atomique des cycles présents dans les graphes moléculaires.

2.2.2 Sous-graphe commun maximum

Nous nous sommes finalement intéressés aux méthodes de recherche de sous-graphe commun maximum, notamment de sous-graphe induit commun maximum (MCIS) et de sous-graphe à arêtes communes maximum (MCES). Ces méthodes ont déjà été appliquées aux graphes moléculaires [Raymond, 2002] et aux graphes de cycles [Nouleho ilemo, 2020].

Étant donnés deux graphes G_1 et G_2 , un sous-graphe commun maximum $G_{1,2}$ de ces graphes est un graphe isomorphe à des sous-graphes de G_1 et G_2 , qui est de taille maximum. $G_{1,2}$ est dit induit ou sommets-maximum (MCIS) s'il contient un nombre maximum de sommets, en conservant toutes les arêtes entre sommets de ce sous-ensemble. Il est dit arêtes-maximum (MCES) s'il contient un nombre maximum d'arêtes. La différence entre le MCIS et le MCES est présentée dans un exemple, figure 5.

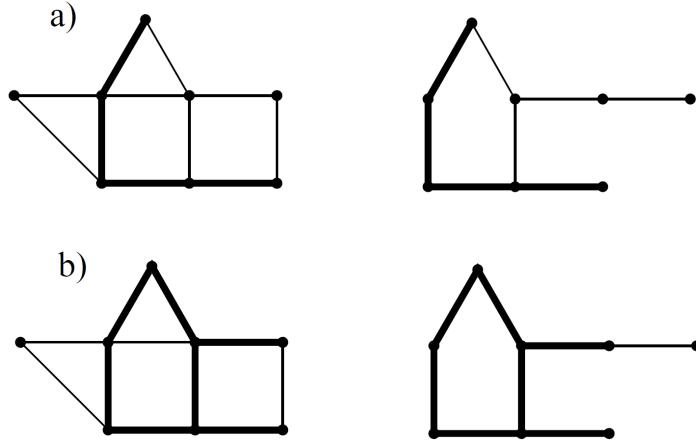


FIGURE 5 – (a) Sous-graphe induit commun maximum (MCIS), (b) Sous-graphe à arêtes communes maximum (MCES).

Bien que les algorithmes basés sur le MCS connaissent plusieurs applications dans le domaine de la chimie et de la biologie, la distinction entre les deux types de MCS n'est pas toujours faite dans la littérature [Ehrlich and Rarey, 2011], rendant difficile la reproductibilité des résultats.

Le problème MCS se réduit au problème de trouver une clique maximum dans un graphe appelé graphe produit de G_1 et G_2 , ce qui en fait un problème NP-complet. Il ne peut donc pas être résolu par un algorithme polynomial (Si $P \neq NP$).

La méthode habituellement employée pour déterminer le MCS de deux graphes est la suivante :

- calculer le graphe produit,
- rechercher une clique maximum dans le graphe produit,
- extraire un sous-graphe commun maximal de G_1 et G_2 .

Soit $G_p = (V, E)$ le graphe produit, également appelé graphe de compatibilité [Ehrlich and Rarey, 2011], de deux graphes $G_1 = (V_1, E_1)$ et $G_2 = (V_2, E_2)$. Il est défini par :

$$V = V_1 \times V_2$$

Soient deux sommets (i_1, i_2) et (j_1, j_2) de G_p . Les sommets i_1 et i_2 (respectivement j_1 et j_2) doivent alors avoir la même étiquette, et sont dits compatibles. Il existe une arête entre (i_1, i_2) et (j_1, j_2) si et seulement si l'une des conditions suivantes est vraie :

- $(i_1, j_1) \in E_1$ et $(i_2, j_2) \in E_2$
- $(i_1, j_1) \notin E_1$ et $(i_2, j_2) \notin E_2$

En outre, la sélection des arêtes du graphe produit peut-être plus amplement contrainte en obligeant les arêtes $[i_1, j_1]$ et $[i_2, j_2]$ à avoir la même étiquette.

Dans le cas d'un MCES, le graphe produit est calculé à partir des linegraphes de G_1 et G_2 . Chaque sommet du linegraphe correspond à une arête du graphe G . De plus, il y a une arête entre deux sommets a_1 et a_2 du linegraphe si et seulement si les arêtes auxquelles ils sont associés, sont incidentes au même sommet de G .

Pour le MCIS, le graphe produit est obtenu directement à partir de G_1 et G_2 .

La recherche d'une clique maximum est l'étape limitante de l'algorithme, puisque son temps de calcul est exponentiel en la taille du graphe produit dans le pire des cas. Un algorithme souvent utilisé est celui de Bron-Kerbosh [Bron and Kerbosch, 1973]. Il applique une recherche exhaustive sur les cliques maximales du graphe de manière récursive. Des heuristiques plus efficaces ont été développées par la suite [Ehrlich and Rarey, 2011], mais l'algorithme de Bron-Kerbosh a l'avantage d'être simple à mettre en oeuvre.

Le MCIS ou le MCES des graphes G_1 et G_2 peut être directement donné par la clique maximum ainsi obtenue. Toutefois, en tenant compte des sommets non connectés dans le graphe produit, le MCS calculé peut ne pas être connexe. On dit qu'il est déconnecté.

Une fois le sous-graphe commun G_{12} déterminé, le coefficient de similarité entre G_1 et G_2 s'obtient en calculant :

$$\text{sim}(G_1, G_2) = \frac{(|V_{12}| + |E_{12}|)^2}{(|V_1| + |E_1|) \times (|V_2| + |E_2|)}$$

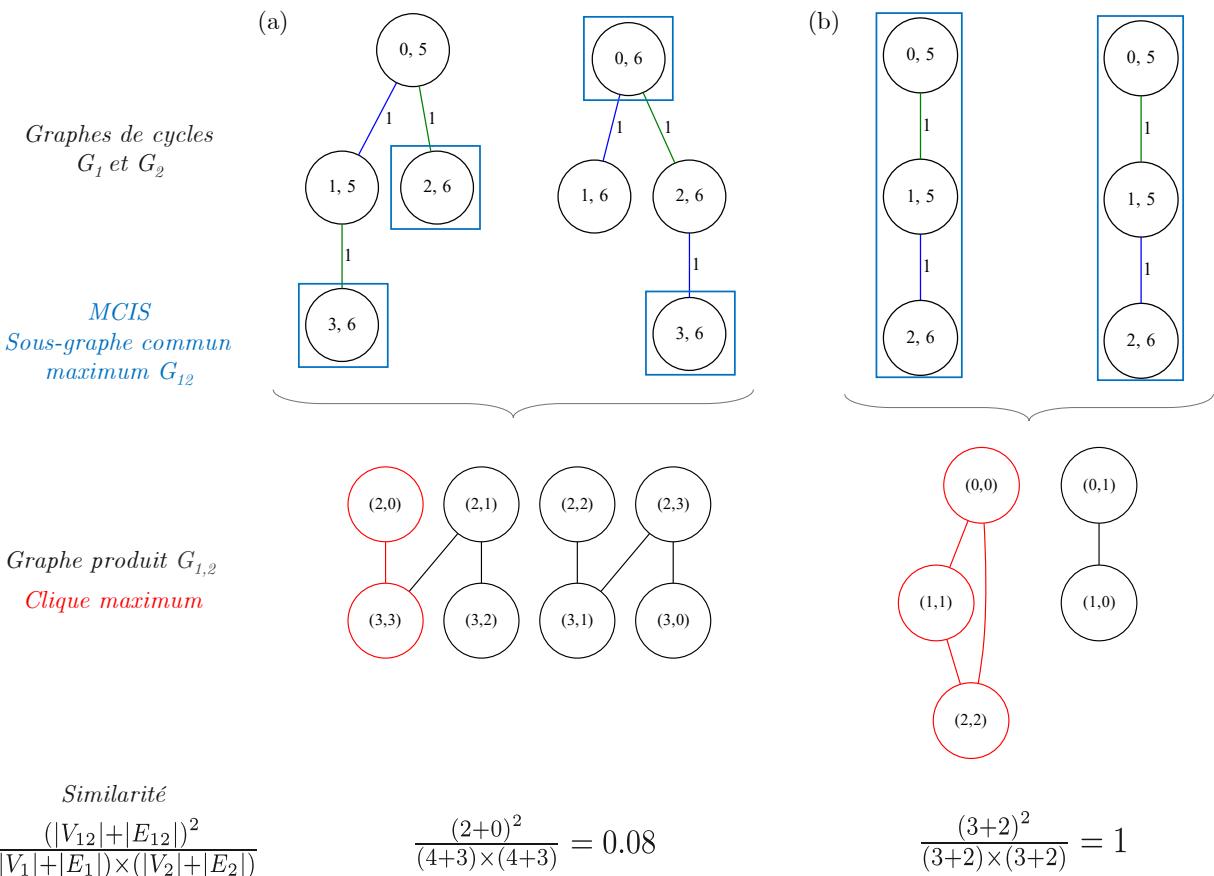


FIGURE 6 – (a) Exemple de MCIS non connexe, (b) Exemple de MCIS connexe.

Sur l'exemple de la figure 6, la similarité des graphes de cycles est calculée par comparaison basée sur le MCIS. Les sommets des graphes produits $G_{1,2}$ sont représentés par le numéro du sommet du graphe

G_1 (en premier) et du graphe G_2 (en deuxième) desquels ils sont issus. En 6a, on observe que le MCIS est déconnecté, et que les graphes sont différents : ils ont un coefficient de similarité de 0,08. Tandis qu'en 6b, le MCIS est connexe et correspond à l'identité des deux graphes. Les graphes G_1 et G_2 ont une similarité de 1, ce qui est cohérent avec le fait qu'ils soient identiques.

La littérature ne semble pas favoriser un des algorithmes de MCS plus que l'autre. Cependant, les publications plus récentes se tournent davantage vers le MCES. Il offrirait peut-être une mesure de similarité plus adaptée aux comparaisons topologiques. Nous avons fait le choix de nous servir de l'algorithme du MCIS pour calculer la similarité des graphes de cycles des molécules. En effet, la structure d'une molécule est déjà capturée par la représentation en graphe de cycles. Néanmoins, il aurait été intéressant de comparer les résultats obtenus avec le MCIS à ceux du MCES.

3 Matériels et Méthodes

Dans cette section, nous présentons les méthodes employées afin d'implémenter chacun des algorithmes nécessaires à la récupération des données moléculaires, la conversion en graphes moléculaires, l'extraction d'une base de cycles minimales, la conversion en graphe de cycles et la comparaison de ces derniers. Nous présentons également les structures de données utilisées ainsi que les outils d'analyse des résultats dont nous nous sommes servis.

L'intégralité du code réalisé au cours de ce projet est accessible en ligne¹.

3.1 Récupération et extraction des données

Le point de départ de notre démarche de récupération des données a été la base de données ChEBI (Chemical Entities of Biological Interest) [Hastings et al., 2016], une ressource incontournable dans le domaine de la chemoinformatique, qui fournit des informations structurées sur les entités moléculaires.

Nous avons utilisé une archive spécifique, *ChEBI_lite_3star.sdf.gz*², contenant des données sur les molécules au format SDF, un standard dans le domaine de la chimie pour représenter des informations moléculaires de manière structurée.

Notre processus de récupération des données s'est articulé autour de plusieurs étapes clés, implémentées en Python. Dans un premier temps, nous avons téléchargé et décompressé l'archive SDF mentionnée ci-dessus. Ensuite, nous avons utilisé la bibliothèque RDKit pour charger et traiter ces données. RDKit est une boîte à outils puissante en chemoinformatique, permettant de manipuler et analyser des structures moléculaires.

Pour chaque molécule chargée à partir de l'archive SDF (Certaines molécules ne sont pas chargées car leur fichier SDF est erroné), nous avons appliqué une phase de pré-traitement pour garantir leur adéquation avec notre étude. Les opérations effectuées incluent l'exclusion des molécules n'étant constituées d'aucun cycle, et la suppression récursive des atomes ayant un seul voisin. Ce processus de nettoyage visait à simplifier les structures moléculaires, en se concentrant sur les éléments essentiels, pour notre étude de comparaison basée sur les graphes de cycles. Les graphes moléculaires ainsi simplifiés sont appelés graphes moléculaires réduits.

Sur les 51 888 molécules initialement chargées, nous obtenons 35 678 molécules après application de

1. Disponible à URL : <https://github.com/NoeDemange/M2AmisProjet>
2. Disponible à URL : <https://ftp.ebi.ac.uk/pub/databases/chebi/>

nos critères de sélection. Les molécules retenues présentent une diversité structurelle, avec un nombre d'atomes allant de 3 à 446, la moyenne étant de 18,23 atomes par molécule (fig 7b). Cette sélection rigoureuse garantit que notre ensemble de données est à la fois pertinent et gérable pour les analyses de comparaison que nous envisageons.

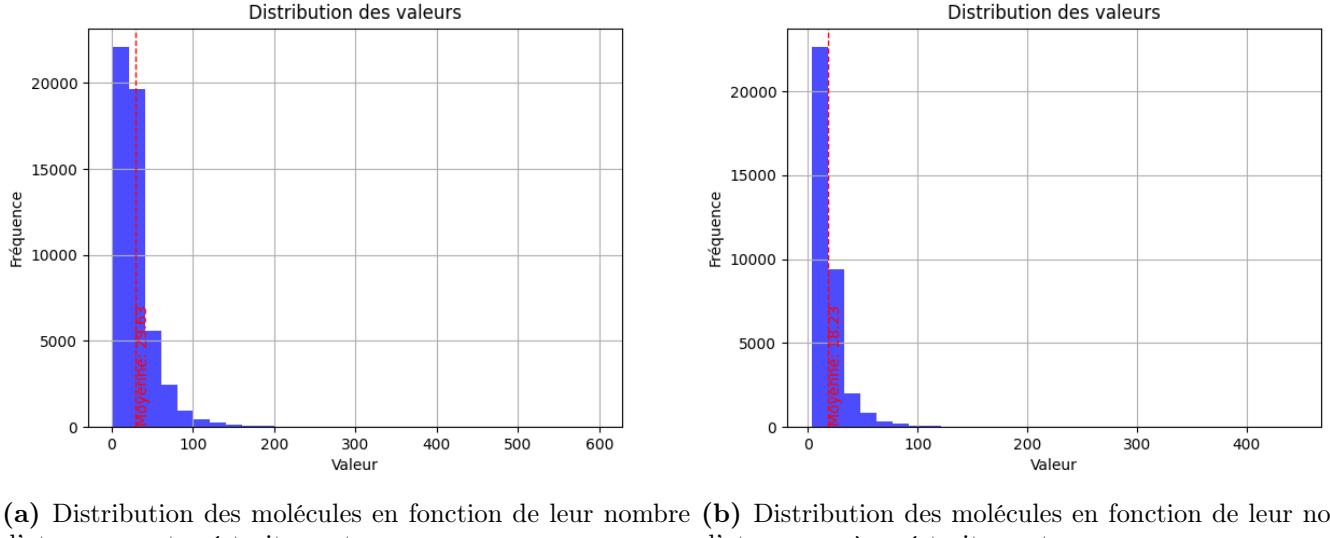


FIGURE 7 – Histogrammes

A l'issue de la phase d'extraction des données, les informations que nous avons jugées essentielles sont conservées dans des fichiers individuels, pour chacune des molécules récupérées. L'identifiant unique que la base ChEBI attribue à chaque molécule, ainsi que la taille de la molécule réduite, sont stockés dans le nom du fichier. La taille des molécules réduites donnant directement celle des graphes moléculaires, la taille maximale des éléments traités peut être récupérée facilement. Une partie de la mémoire tampon peut ainsi être pré-allouée afin d'augmenter la rapidité des opérations. En effet, le reste du programme est écrit en C pour garantir une certaine rapidité de traitement. Dans les fichiers moléculaires se trouvent la matrice d'adjacence (triangulaire) des graphes moléculaires réduits, ainsi que la chaîne de caractères donnant le type d'atome associé à chaque sommet.

3.2 Structures de données

Nous avons choisi d'utiliser des matrices d'adjacence pour stocker les différentes représentations en graphe utilisées au cours de notre programme. Ce choix est motivé par la facilité d'implémentation et d'utilisation de telles structures. Bien que les matrices d'adjacence des graphes moléculaires et de leur graphe de cycles soient très creuses, les résultats de l'analyse préliminaire présentés en figure 7b confortent dans l'idée que la perte en espace mémoire est négligeable. En outre, de part le traitement séquentiel des molécules, la mémoire de la structure stockant le graphe moléculaire est immédiatement libérée après la création du graphe de cycles.

Un cycle est représenté par un tableau contenant le numéro des sommets qui le composent. L'ordre d'addition des sommets est important : le premier sommet correspond au sommet "source" à partir duquel le cycle a été trouvé (la racine du parcours, dans la section 3.3), puis les sommets suivants suivent l'un des deux sens de parcours du cycle jusqu'au retour sur le sommet source.

3.3 Base de cycles minimale canonique d'un graphe

Après lecture des fichiers moléculaires, nous enregistrons l'identifiant unique de cette molécule en provenance de la base de données ChEBI, sa formule sous forme d'une chaîne de caractères, ainsi que son nombre d'atomes et la matrice d'adjacence de son graphe moléculaire dans une structure grapheMol représentant un graphe moléculaire.

Plutôt que d'implémenter l'algorithme de McKay qui permet de garantir la propriété canonique de la base de cycles, nous avons utilisé la bibliothèque *nauty* dans laquelle l'algorithme est programmé de manière optimisée. Comme précisé dans la section 2.1.2, cet algorithme s'effectue en temps polynomial pour des graphes moléculaires. Grâce à la fonction *densenauty*, nous obtenons la numérotation canonique des sommets d'un graphe.

Une permutation est ensuite effectuée sur les lignes de la matrice d'adjacence, puis sur ses colonnes, conformément à la numérotation canonique des sommets.

L'implémentation de l'algorithme de Horton se base sur les méthodes décrites dans la thèse de Stefi Nouleho [Nouleho ilemo, 2020] ainsi que dans l'article de Vismara [Vismara, 1997]. Pour chaque sommet du graphe moléculaire canonique, un parcours en largeur est effectué afin de trouver les couples de sommets avec lesquels il forme un cycle. En effet, dans les graphes où toutes les arêtes ont le même poids, le parcours en largeur donne un plus court chemin entre la racine et chaque sommet atteint.

Pour chaque nouveau sommet rencontré dans l'arbre du parcours en largeur depuis un sommet racine x , une vérification est effectuée pour savoir si le sommet a déjà été atteint. Pour ce faire, un tableau *visités* de la taille du graphe est initialisé à 0 pour tous les sommets. A chaque fois qu'un nouveau sommet est vu, la case du tableau correspondante est passée à 1.

Un tableau *parents*, également de la taille du graphe, mémorise le prédécesseur de chaque sommet dans l'arbre de parcours. Si un sommet y a déjà été vu, cela signifie que l'arête formée par ce sommet et son prédécesseur z appartient à un cycle. Mais ce cycle ne contient peut-être pas le sommet racine x . Un test est donc réalisé pour savoir si la chaîne entre la racine et y , et celle entre la racine et z , sont arête-disjointes. Cela peut se vérifier simplement en connaissant le niveau des sommets y et z dans l'arbre de parcours. Le niveau d'un sommet correspond au nombre d'arêtes le séparant de la racine. Ainsi, tous les voisins de la racine sont au niveau 1. Or, deux sommets qui sont reliés par une arête dans le graphe moléculaire ne peuvent pas être atteints à plus d'un niveau d'intervalle. Il suffit donc de comparer deux-à-deux les prédécesseurs de y et z qui sont situés au même niveau sur la chaîne les reliant à x .

Il faut également vérifier que le cycle n'a pas déjà été ajouté à la liste. Une solution facile est d'insérer le cycle que lorsque le sommet en racine a un numéro supérieur à tous les autres sommets du cycle. Comme chaque cycle ne possède qu'un seul sommet de numéro supérieur à tous les autres, il sera ajouté au plus une fois.

Après avoir obtenu la liste des cycles du graphe, nous réalisons un tri fusion pour les classer dans l'ordre croissant de leur nombre de sommets. Nous avons choisi d'utiliser l'algorithme du tri fusion, car il a la meilleure complexité moyenne en $O(n \log(n))$. Enfin nous effectuons une élimination gaussienne binaire sur cette ensemble de cycles, pour trouver la base de cycles minimale de ce graphe.

L'élimination de Gauss est réalisée sur une matrice d'incidence décrivant quelles arêtes appartiennent à quels cycles. L'élimination gaussienne consiste à comparer chaque ligne de la matrice aux lignes qui la succèdent et de chercher à voir si elles sont linéairement indépendantes. Comme une ligne correspond à un cycle, cela revient à déterminer si un cycle peut être obtenu en faisant la somme disjointe d'autres

cycles deux-par-deux. Pour accélérer le processus, on utilise le fait que les cycles sont triés par ordre de taille croissant. Ainsi, un cycle est toujours comparé aux cycles de numéro supérieur dans la matrice. Si la première arête du cycle qui est en train d'être comparé aux autres, appartient également à un des cycles c_i qui suit, le XOR des deux vient remplacer la ligne du cycle c_i dans la matrice. Si à l'issue de l'élimination, la ligne d'un cycle contient uniquement des 0, cela signifie qu'il n'est pas linéairement indépendant des autres. Les cycles restant sont donc ceux d'une unique base de cycles minimale.

La figure 8 donne la liste de cycles obtenus pour la molécule de quinine après l'application du parcours en largeur sur tous les sommets. Cette liste contient un cycle supplémentaire par rapport à la base de cycles minimale extraite.

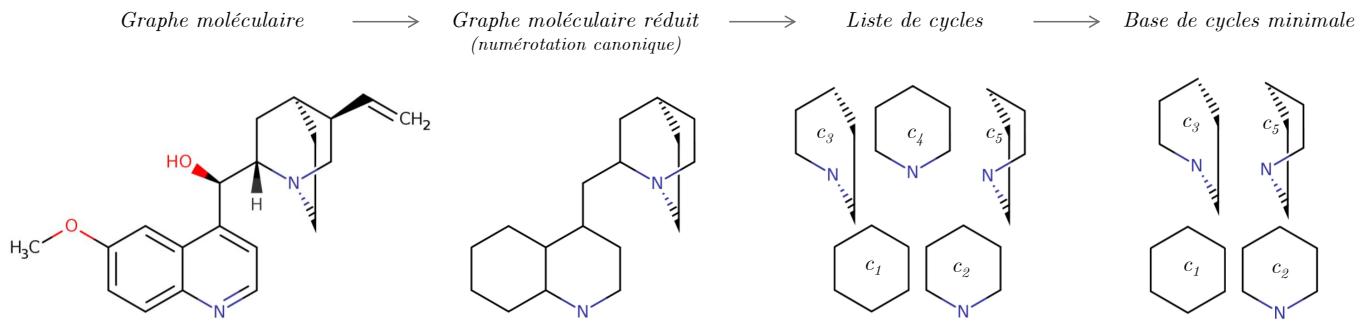


FIGURE 8 – Exemple de liste de cycles et de base de cycles minimale obtenues avec notre procédure, pour la molécule de quinine.

3.4 Transformation en graphe de cycles

Passer de la liste de cycles au graphe de cycles n'est pas une opération complètement triviale. En revanche, certaines étapes se font facilement. Par exemple, les sommets du graphe sont directement obtenus à partir des cycles et de leur taille.

Les arêtes de type 1 (d'après la modélisation présentée en section 2.1.1), entre les cycles possédant des sommets en commun sont également simples à déterminer : il suffit de trier les tableaux contenant les sommets des cycles, puis de les comparer deux-à-deux et de compter le nombre de sommets communs aux cycles. L'arête de type 1 doit être étiquetée par le nombre d'arêtes partagées par les cycles dans le graphe moléculaire. Ce chiffre s'obtient en faisant $nb_sommets - 1$, avec $nb_sommets$ le nombre de sommets partagés par les cycles.

Les arêtes de type 2 sont plus difficiles à trouver : être capable de dire si deux cycles sont disjoints est facile, mais il est plus compliqué de savoir s'ils sont reliés par une chaîne qui n'appartient à aucun cycle. Dans un graphe connexe, une arête n'appartient à aucun cycle si et seulement si c'est un isthme. Il faut donc trouver les isthmes du graphes moléculaires. Pour chaque isthme i , on peut regarder s'il est incident à un sommet appartenant à un cycle c_d . Si tel est le cas, en mémorisant le cycle de départ, il est possible de faire un parcours en profondeur sur les arêtes voisines de i qui sont également des isthmes. Dès qu'une arête est incidente à un sommet contenu dans un cycle c_a , une arête de type 2 peut être ajoutée dans le graphe de cycles entre les cycles c_d et c_a . Cette méthode nécessite de connaître pour chaque sommet du graphe moléculaire la liste des cycles auxquels il appartient. Un tableau peut être utilisé à cette fin.

Pour trouver les isthmes, il est possible de marquer les arêtes qui appartiennent aux cycles de la base minimale. On peut noter que ce marquage doit être effectué avant de chercher les arêtes de type 1, car

l'ordre des sommets des cycles n'est sinon pas conservé. Toutes les arêtes non marquées restantes sont des isthmes, autrement elles appartiendraient à un cycle, et ce cycle aurait dû être présent dans la base minimale. Ce marquage peut être réalisé en associant un nombre différent aux arêtes marquées dans la matrice d'adjacence du graphe.

Enfin, une fois un isthme traité lors du parcours en profondeur sur les arêtes, il doit être marqué comme tel. En effet, la condition d'arrêt du parcours est de ne plus avoir d'isthmes non marqués.

A l'issue de cette procédure, nous obtenons une représentation canonique du graphe de cycles d'un graphe moléculaire.

3.5 Comparaison des graphes de cycles moléculaires

Afin de comparer les graphes de cycles, nous appliquons tout d'abord un calcul de similarité basé sur le MCIS. Puis une distance de Levenshtein est calculée sur les chaînes de caractères donnant le type des atomes. Enfin, les deux mesures sont combinées en les multipliant.

3.5.1 MCIS

La comparaison de deux graphes G_1 et G_2 basée sur le MCIS se déroule en deux grandes étapes : une première étape consiste à déterminer le graphe produit $G_{1,2}$ des deux graphes à comparer, et une deuxième à trouver une clique maximum dans le graphe produit. Le sous-graphe commun est directement extrait de la clique maximum, par comparaison à G_1 ou G_2 , en veillant à ne conserver que les arêtes de la clique qui sont associées à une arête dans les deux graphes.

Nous utilisons à nouveau une matrice d'adjacence pour stocker les informations du graphe produit. Il faut tout d'abord déterminer la liste des sommets des graphes G_1 et G_2 qui sont compatibles entre eux. Deux sommets sont compatibles s'ils ont la même étiquette. Cela signifie que les sommets du graphe produit sont l'ensemble des couples de cycles de même taille formés d'un cycle de G_1 et de G_2 . Puis les arêtes sont ajoutées dans la matrice d'adjacence selon la définition du graphe produit donnée dans la section 2.2.2.

Concernant l'algorithme de calcul de clique maximum, nous avons choisi de reprendre l'algorithme utilisé par [Nouleho ilemo, 2020], qui est une version adaptée de Bron-Kerbosch. Cet algorithme énumère les cliques maximales d'un graphe de manière récursive. Il peut donc être utilisé pour trouver une clique maximum. Il est également assez facilement transposable en programme. De plus, Stefi Nouleho propose une version récursive qui utilise peu de structures intermédiaires. Dans l'algorithme de Bron-Kerbosch, une liste de sommets candidats P est initialisée avec tous les sommets du graphe. Une liste X sert à mémoriser les sommets déjà traités, et une liste R contient les cliques en construction. A chaque itération, on regarde si la liste P n'est pas vide, auquel cas l'algorithme fait un appel récursif en ajoutant le premier sommet candidat v à R , et les ensembles P et X sont restreints aux sommets voisins de v , s'ils en contiennent. Quand P est vide, il n'y a plus aucun sommet candidat. Si X est également vide, alors R contient une clique maximale. Sinon X contient encore des sommets, et toutes les cliques maximales qui contiennent potentiellement v ont déjà été traitées. Dans ce cas, l'algorithme revient sur ses pas. Pour trouver une clique maximum, il faut en plus enregistrer la plus grande clique maximale rencontrée au fur et à mesure de l'exécution de l'algorithme.

Toutefois, en plus d'avoir une complexité exponentielle en la taille de l'entrée, cet algorithme peut s'avérer très inefficace pour les graphes contenant beaucoup de cliques maximales. Il existe une version "avec pivot", où le choix du sommet candidat peut-être fait de sorte à minimiser le nombre d'appels

récursifs. Une amélioration de la méthode de calcul de clique maximum pourrait donc être d'utiliser plutôt la version pivot de Bron-Kerbosch.

Un exemple de résultat de graphe produit et de clique maximum est donné figure 9. Les deux sommets de la clique correspondent à des couples de cycles disjoints dans les graphes de cycles. Le sous-graphe commun est donc déconnecté. Il s'agit du même exemple que celui présenté dans la figure 6a. La figure 9 montre en plus les résultats intermédiaires obtenus lors de la comparaison des molécules de ChEBI ID 47 et 109, en particulier les graphes moléculaires réduits et les graphes de cycles correspondants.

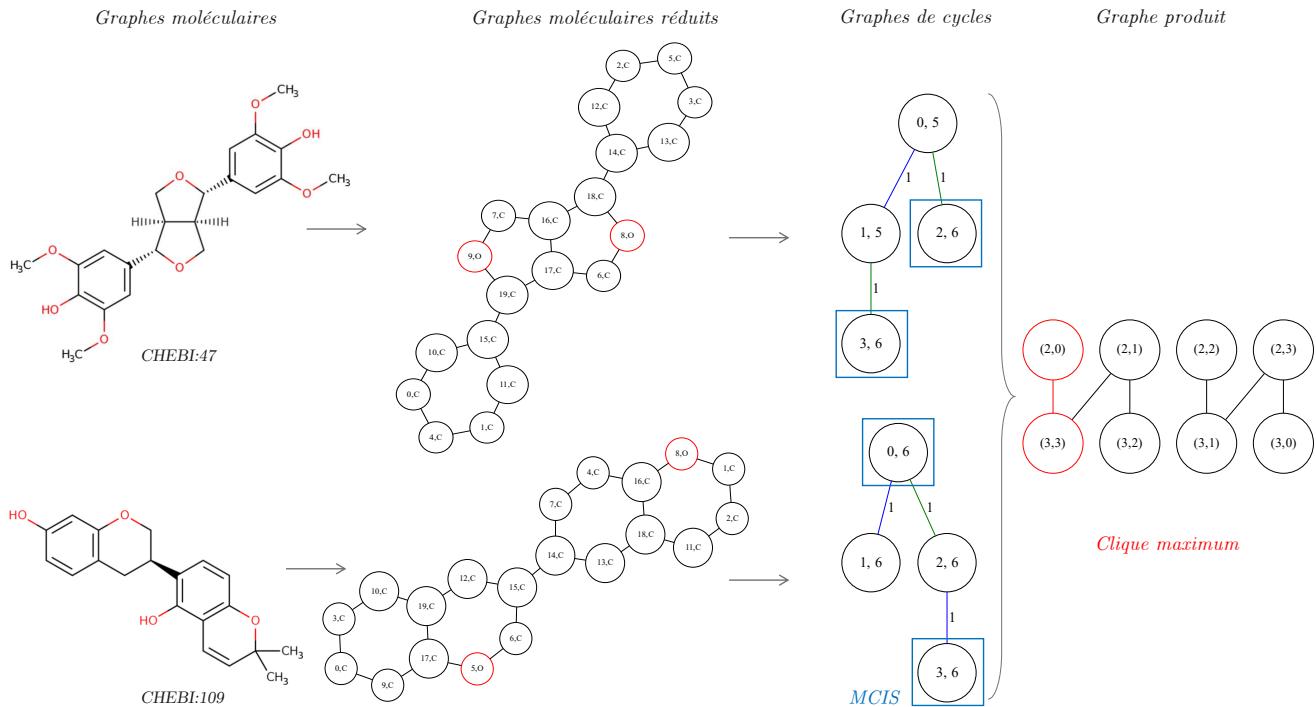


FIGURE 9 – Exemple de résultats de comparaison sur deux molécules jugées "différentes".

3.5.2 Levenshtein

Dans cette section, nous allons parler de la façon dont nous avons implémenté la similarité de Levenshtein et des difficultés auxquelles nous avons été confrontés.

Comme expliqué dans la section 2.2.1.2, nous cherchons à comparer des chaînes de caractères représentant le symbole des atomes. L'un des problèmes auxquels nous avons dû faire face, est celui des symboles des atomes pouvant contenir deux lettres comme par exemple "Fe" pour le Fer. Cela nous oblige en langage C à stocker un symbole dans un tableau de type 'char' et donc l'ensemble des symboles dans un tableau à double entrée de 'char'. Nous sommes donc contraints à utiliser la fonction *strcmp* pour comparer deux symboles.

L'autre problème que nous avons rencontré, est sur la performance de notre implémentation car cet algorithme est appelé à chaque comparaison soit 499500 fois pour la comparaison de 1000 molécules. Dans l'algorithme, nous utilisons deux tableaux d'entiers pour conserver les distances calculées jusqu'à cette étape, que nous avons appelés *precedent* et *courant*. Pour éviter de devoir allouer de la mémoire pour ces tableaux à chaque fois que nous appelons cette fonction, nous initialisons des buffers pour réservrer de l'espace mémoire pour cette fonction.

Ainsi notre implémentation de l'algorithme commence par l'initialisation du tableau de *precedent* avec chaque case ayant la valeur de son indice. Puis on réalise une double boucle (une de la taille de la plus petite chaîne et l'autre de la plus grande) qui nous permet de parcourir chaque symbole d'atome de la plus petite chaîne et de les comparer à ceux de la plus grande. Donc nous comparons les deux caractères, si les deux sont identiques, on obtient un coût de 0 et sinon un coût de 1. Puis nous mettons à jour le tableau *courant* en position *j* avec le minimum entre *precedent[j]+1* (effacement du caractère de la chaîne 1), *courant[j-1]+1* (insertion dans chaîne 2 du nouveau caractère de chaîne 1) et *precedent[j-1]+coût* (substitution de caractère). À la fin nous récupérons la valeur de la distance entre les deux chaînes de caractères dans la dernière case du tableau *courant*. Pour obtenir la similarité, nous soustrayons à 1 cette distance puis la normalisons en la divisant par la taille de la plus grande chaîne.

Algorithm 1 Calcul de la distance de Levenshtein

```

1: function DISTANCELEVENSTEIN(mol1, taille1, mol2, taille2, matrice)
2:   precedent ← matrice[0], courant ← matrice[1]
3:   échanger chaine1 et chaine2 si nécessaire pour que taille1 ≤ taille2
4:   for j de 0 à taille2 inclus do
5:     precedent[j] ← j
6:   end for
7:   for i de 1 à taille1 inclus do
8:     courant[0] ← i
9:     for j de 1 à taille2 inclus do
10:      cout ← (chaine1[i - 1] = chaine2[j - 1]) ? 0 : 1
11:      courant[j] ← min(precedent[j] + 1, courant[j - 1] + 1, precedent[j - 1] + cout)
12:   end for
13:   copier courant dans precedent
14: end for
15: distance ← courant[taille2]
16: if matrice = NULL then
17:   libérer precedent, libérer courant
18: end if
19: return distance
20: end function

```

Exemple sur les molécules Chebi : 33202 et 63186 (fig : 10)

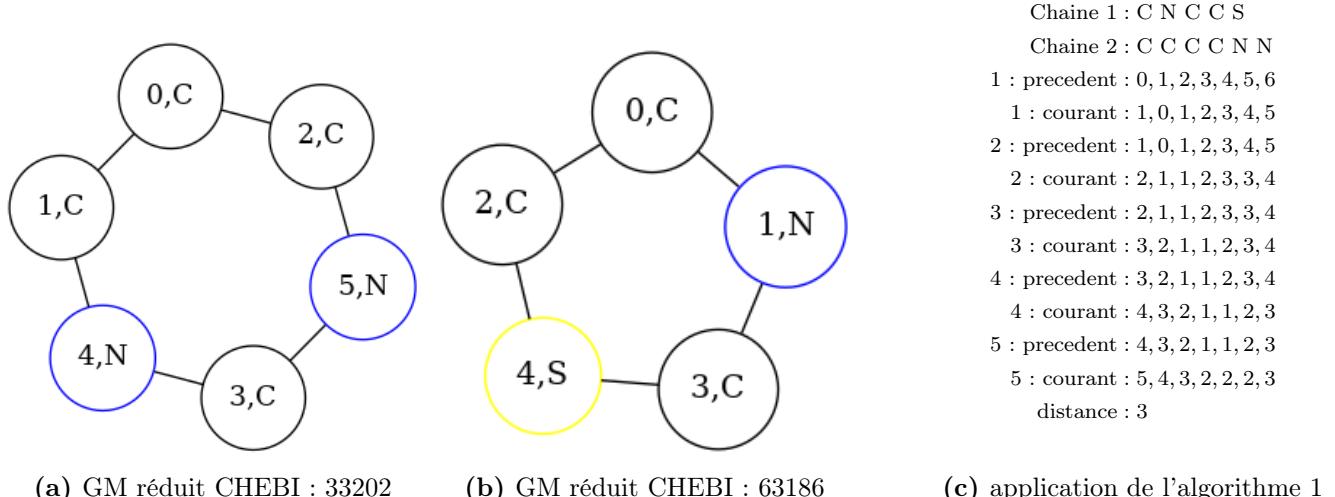


FIGURE 10 – Exemple de calcul de distance de Levenshtein entre les molécules 33202 et 63186

3.6 Analyse des Similarités

Pour mener à bien l'analyse des similarités entre les molécules, nous utilisons le langage R et l'application Shiny *OrderedHeatMapsAnalysis (OMHA)*³. L'objectif de cette section est d'identifier des groupes de similarité ou des classes d'équivalence parmi les molécules analysées, en générant une carte de chaleur (heatmap) segmentée selon les groupes identifiés.

3.6.1 Calcul des Distances

À partir de notre code, nous générerons une matrice de similarité (où 1 représente une similarité parfaite et 0 une différence totale). Pour effectuer la classification hiérarchique (voir section 3.6.2), nous avons besoin d'une matrice de distance (où 0 indique une similarité parfaite). Pour ce faire, nous proposons deux méthodes. La première consiste à obtenir la matrice de distance en prenant le complément à 1 de la matrice de similarité. Cette méthode conserve au mieux les données initiales. La seconde méthode consiste à calculer la distance euclidienne entre les lignes de notre matrice selon la formule : $\sqrt{\sum_i(x_i - y_i)^2}$. Cette méthode, largement utilisée, est simple à implémenter et nous permet de faire une distance entre deux molécules par rapport à leur distance avec toutes les molécules étudiées.

3.6.2 Classification Hiérarchique

Ensuite, nous effectuons une classification hiérarchique sur la matrice de distance. Pour ce faire, nous utilisons la méthode "ward.D2" [Murtagh and Legendre, 2014] de la fonction *hclust* de R. Cette méthode est une méthode de classification ascendante hiérarchique.

Nous réorganisons ensuite les branches de la classification hiérarchique en utilisant la fonction *DendSer* [Gruvaeus and Wainer, 1972] du package *DendSer*. Cette réorganisation repose sur la matrice de distance et utilise la méthode de coût "costARc".

De plus, nous appliquons la fonction *cutreeHybrid* du package *DynamicTreeCut* [Langfelder et al., 2008]. Cette fonction permet de détecter des clusters dans une classification hiérarchique en se basant sur la forme des branches. L'avantage de cette méthode est qu'elle ne requiert pas de spécification préalable du nombre de clusters en argument de la fonction.

3.6.3 Heatmap

Pour visualiser à la fois l'ordre généré par la classification hiérarchique, les clusters ainsi que les valeurs de similarité entre toutes les molécules, nous utilisons le package *complexHeatmap* [Gu et al., 2016, Gu, 2022].

4 Résultats et Analyse

Dans cette section, nous allons aborder les différents résultats obtenus, et les analyser avec les outils présentés plus haut. Tout d'abord, nous examinerons notre choix de métrique de distance. Ensuite, nous comparerons l'efficacité de l'utilisation de la similarité liée au MCIS seule, la similarité obtenue par la

3. Application de G. Sapriel, N. Demange.

Disponible à l'adresse : <https://github.com/NoeDemange/OrderedHeatMapAnalysis>

distance de Levenshtein, ainsi que la combinaison des deux approches. Enfin, nous procéderons à une analyse des classes d'équivalence résultantes de la comparaison de 1000 molécules.

4.1 Examen de la sélection de la métrique de distance

Nous commençons par comparer les deux méthodes de calcul d'une matrice de distance. Comme expliqué dans la section 3.6.1, nous devons choisir une façon de convertir la matrice de similarité en matrice de distance. Cette seconde représentation nous permettra de rassembler les graphes de cycles les plus "proches" en cluster, grâce à l'utilisation d'une classification ascendante hiérarchique.

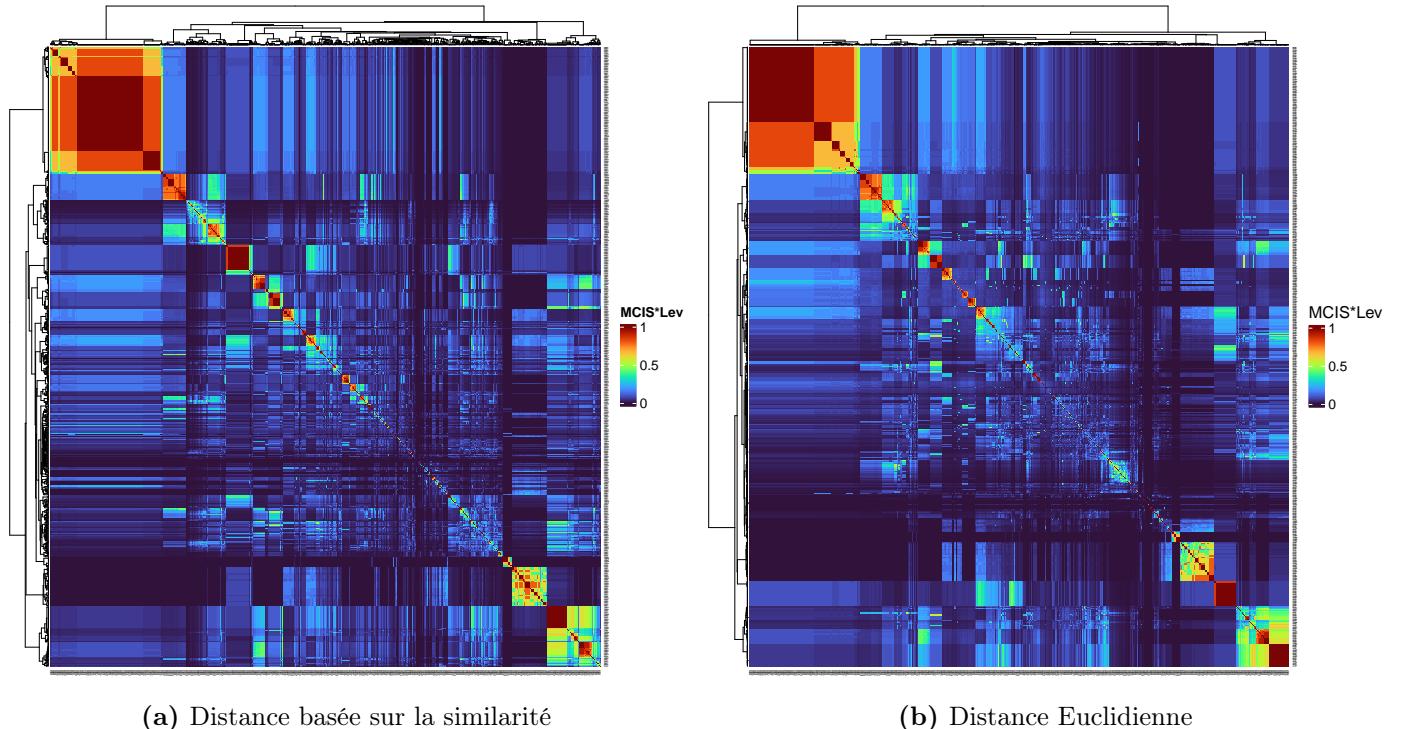


FIGURE 11 – Heatmap de la comparaison de 1000 molécules ordonnée par classification hiérarchique sur deux distances différentes

Lorsque nous transformons la similarité en distance simple, figure 11a), nous observons que les molécules très similaires entre elles sont bien classées, tandis que les molécules moins similaires vont se retrouver réparties entre plusieurs catégories.

Ce phénomène est notamment visible dans la zone supérieure gauche, où les molécules qui sont proches de celles qui forment le grand carré rouge (celles ayant une similarité d'intensité orange), ne forment pas qu'un seul groupe. La distance basée sur la similarité cherche donc avant tout à regrouper correctement les éléments dont la similarité est d'intensité rouge. Ceci s'explique par le fait que nous ayons analysé les graphes de cycles deux-par-deux pour déterminer leur similarité. Cette distance reflète donc les meilleures similarités trouvées par paire, en occultant les similarités que les deux molécules partagent avec les autres.

Au contraire quand nous organisons la matrice à partir de la distance Euclidienne, figure 11b, nous observons cette fois-ci que les molécules proches du grand carré rouge, dans la partie supérieure gauche, sont toutes regroupées et forment un seul groupe d'intensité orange, bien que ces molécules ne soient pas toutes très similaires entre elles. Cette constatation s'explique par le fait que nous regardons cette fois

les similarités des molécules par rapport à l'ensemble des molécules. Cela permet de mettre en évidence que cet ensemble de molécules est proche de celles qui forment le grand carré rouge.

Nous proposons ainsi deux façons d'analyser les classes d'équivalence des molécules. Une première façon permet de trouver des groupes de molécules qui partagent de fortes similarités, sans tenir compte des molécules avec lesquelles elles sont moins similaires, et une seconde méthode permet de trouver les sous-ensembles de molécules similaires à d'autres sous-ensembles de molécules.

Dans la suite des analyses nous utiliserons la distance Euclidienne.

4.2 Analyse des approches de similarité : MCIS, distance de Levenshtein, et combinaison des deux

Nous allons, dans cette section, comparer les trois approches de calcul de similarité que nous avons testées. La première approche repose sur le calcul de la similarité à partir du MCIS appliquée aux graphes de cycles. La deuxième approche se base sur la distance de Levenshtein que nous calculons sur les chaînes de caractères représentant le type des atomes. Ces séquences sont organisées selon l'ordre des sommets trouvé par l'algorithme de McKay, dans les graphes moléculaires. Nous appellerons similarité de Levenshtein la valeur $(1 - \text{distance})$ normalisée. Puis pour finir nous étudierons la combinaison des deux similarités par multiplication.

Nous commençons par l'analyse des résultats obtenus en utilisant la méthode du MCIS. Ces résultats, calculés sur les graphes de cycles des molécules, ont l'avantage de nous présenter une méthode qui permet de tenir compte de la structure (composée de cycles) de la molécule tout en allant plus vite qu'une analyse sur le graphe moléculaire.

Nous pouvons voir sur la figure 12, que les graphes de cycles des molécules ChEBI ID 84251 et 35005 sont très similaires. En comparant ces deux molécules, nous observons que les deux contiennent un seul cycle de taille 6. La comparaison basée sur le MCIS donne donc que des molécules ayant une structure similaire, ont des graphes de cycles similaires, ce qui est le résultat souhaité. Nous pouvons voir également que les molécules d'ID 18413 et 91005 sont respectivement très éloignée et moyennement éloignée, ce qui est cohérent car la molécule 18413 contient seulement un cycle de 5 atomes alors que la molécule 91005 contient deux cycles de taille 6. Nous pouvons encore observer que la molécule 75695 est très éloignée de toutes les autres molécules. De nouveau, ce résultat est cohérent, car il s'agit de la molécule *dC₂₁* qui est un oligonucléotide composé de 21 2'-désoxycytidine. Elle contient donc beaucoup de cycles ce qui l'éloigne énormément des autres.

Finalement, cette méthode de similarité permet de correctement analyser la structure des molécules. Cependant, elle ne donne aucune indication sur les atomes qui composent les cycles.

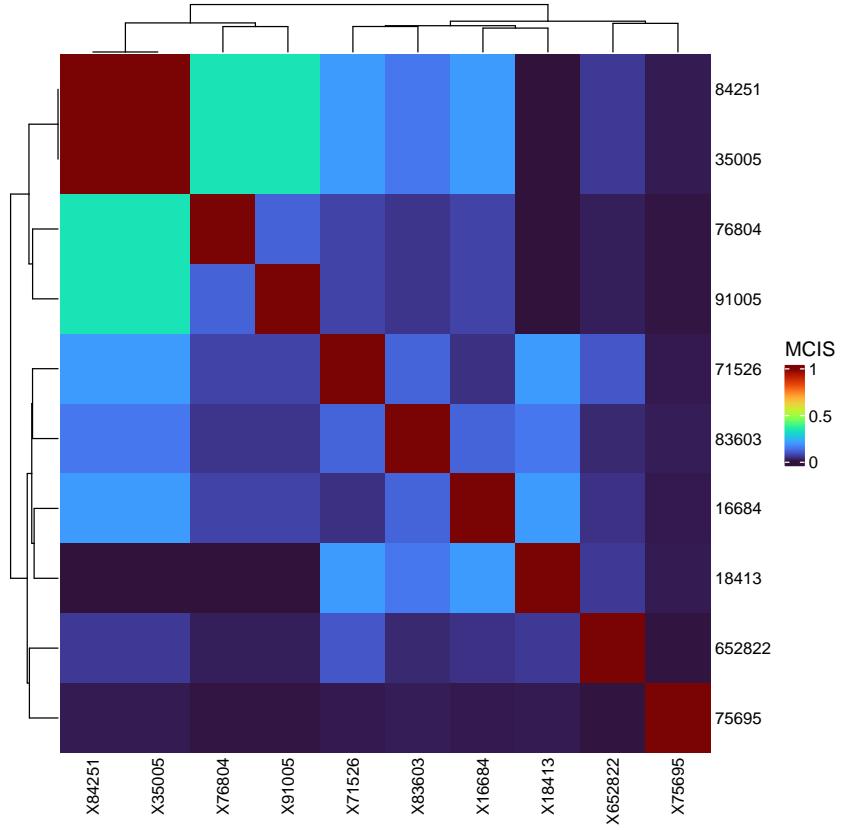


FIGURE 12 – Heatmap de la comparaison de 10 molécules utilisant la méthode du MCIS pour calculer la similarité

Nous avons donc ensuite décidé d'utiliser la similarité de Levenshtein. En se basant sur cette similarité, figure 13), nous observons que les molécules 84251 et 35005 sont toujours très similaires. Cela reste cohérent avec le fait qu'elles possèdent toutes les deux un unique cycle à 6 carbones. En revanche, nous nous apercevons cette fois-ci que la molécule 18413 est proche d'elles car elle contient un unique cycle de 4 carbones et 1 oxygène. La distance de Levenshtein entre les molécules 18413 et 84251 est alors de 2 (une substitution et une insertion/suppression), ce qui les rend similaires selon cette distance. De nouveau, nous pouvons voir que la molécule 75695 est très éloignée de toutes les autres molécules. Cela est toujours dû à sa taille, même si nous pouvons remarquer qu'elle semble un peu moins éloignée d'autres molécules de plus grande taille (par exemple, la molécule de ChEBI ID 76804).

Cette similarité ne tient pas compte de la structure cyclique des molécules, mais elle apporte des indications sur la composition en atomes et la taille de la molécule.

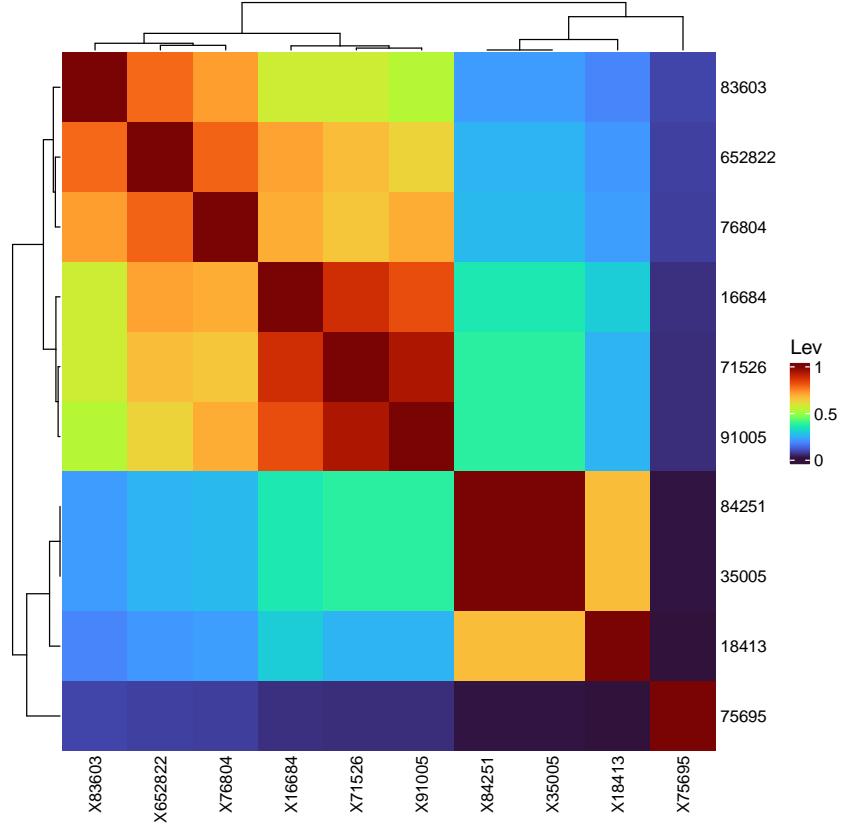


FIGURE 13 – Heatmap de la comparaison de 10 molécules utilisant la similarité de Levenshtein

Les méthodes de comparaison par MCIS et par distance de Levenshtein nous ont paru être complémentaires. Ainsi, nous avons décidé d'analyser le multiple des deux. Sur la figure 14, nous pouvons voir que nous conservons davantage la structure de la heatmap avec seulement le MCIS, mais que les similarités de plus faible intensité sont diminuées. Cela permet d'affiner notre méthode de comparaison et de tenir compte à la fois de la structure cyclique de la molécule et de la composition atomique de cette structure.

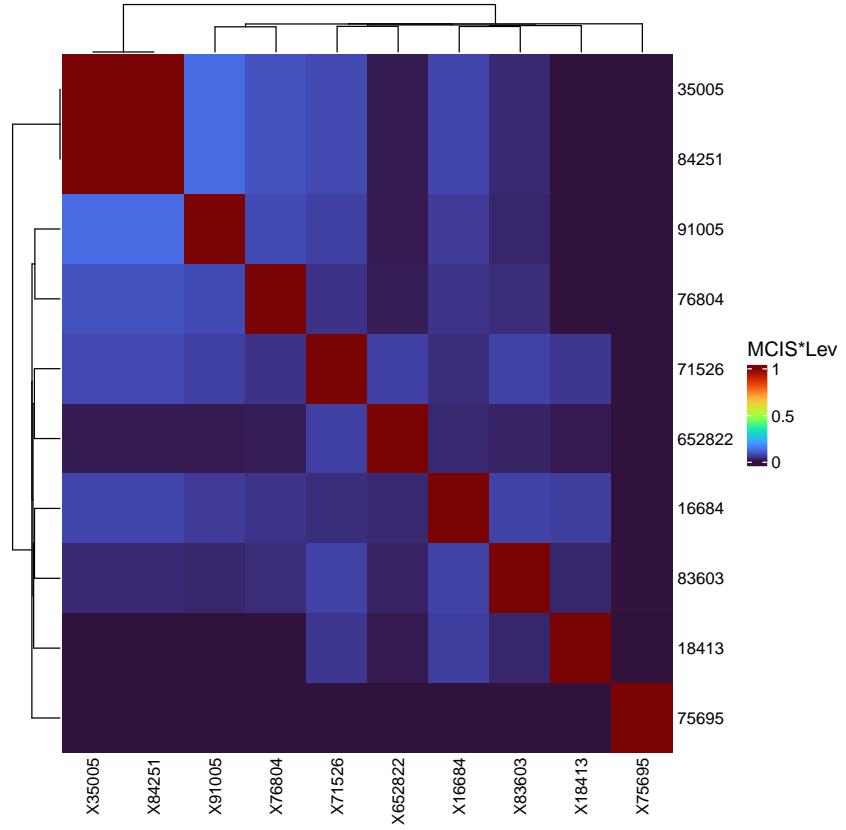


FIGURE 14 – Heatmap de la comparaison de 10 molécules utilisant la similarité par MCIS multipliée par la similarité de Levenshtein

4.3 Analyse des classes d'équivalences

Dans cette section, nous allons analyser les similarités calculées deux à deux pour 1000 molécules. Nous avons choisi de limiter le nombre de molécules comparées, d'une part parce que notre méthode de comparaison utilise un algorithme de complexité exponentielle, et d'autre part pour faciliter la lisibilité des résultats. Les fichiers de molécules sont sélectionnés par ordre croissant de création dans le répertoire de fichier. Nous utilisons la similarité calculée par la multiplication entre les résultats de similarité par MCIS et par distance de Levenshtein.

La figure 15 montre les résultats obtenus. En découplant la classification hiérarchique avec `cutreeHybrid` et un `deepSplit` à 3, nous trouvons 19 clusters de molécules. Le ChEBI ID des molécules composant chaque cluster est disponible en Annexe A.

Dans la suite, nous considérons les clusters numérotés de 1 à 19 en partant du haut.

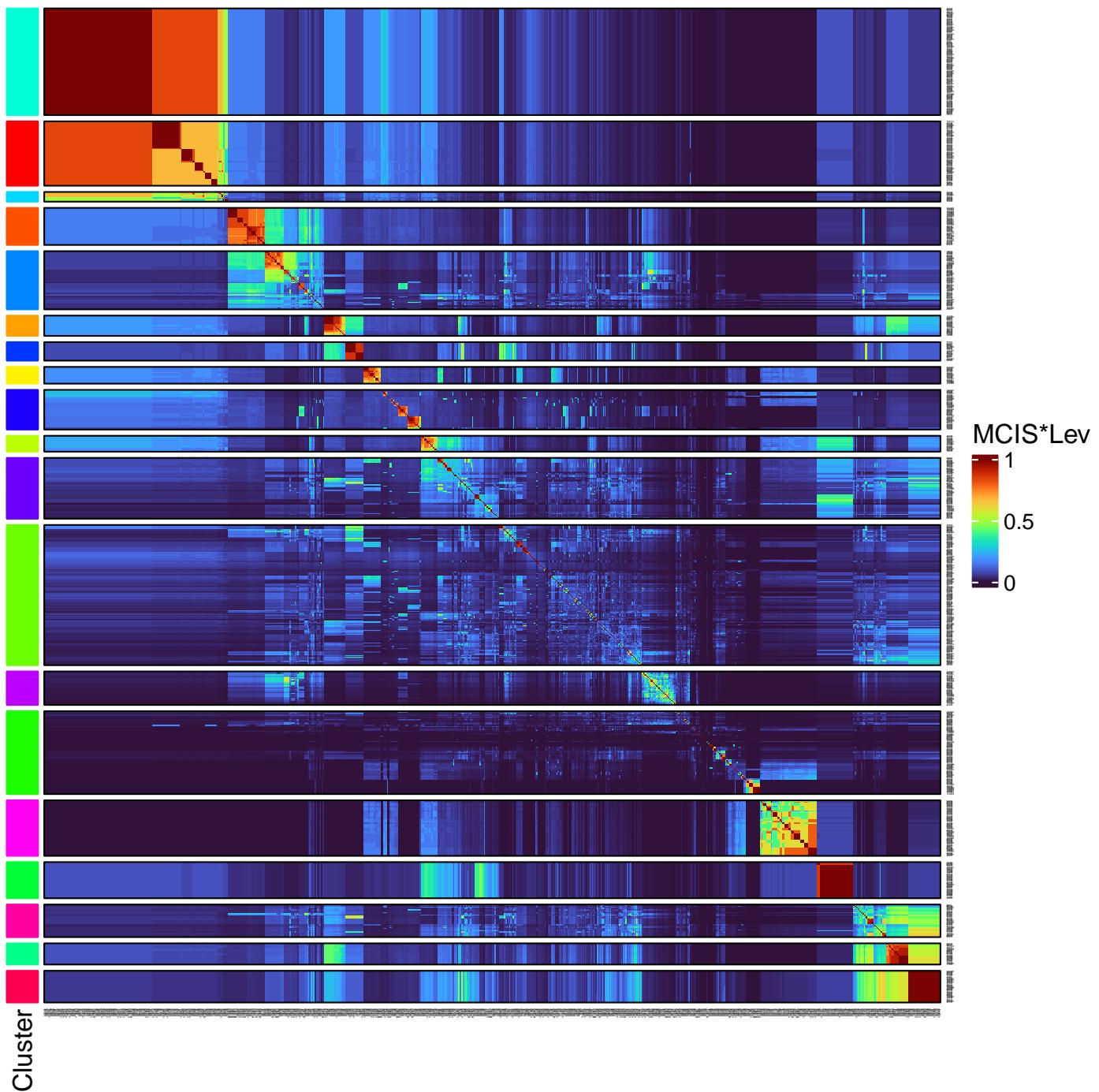
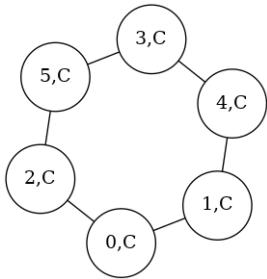
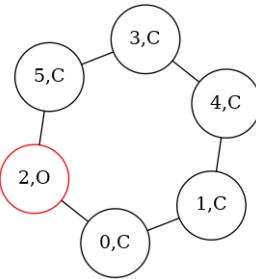


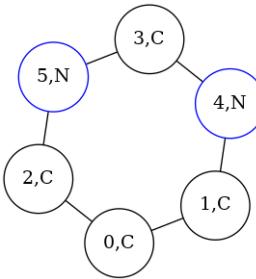
FIGURE 15 – Heatmap de la comparaison de 1000 molécules découpé avec cutreeHybrid et un deepSplit à 3



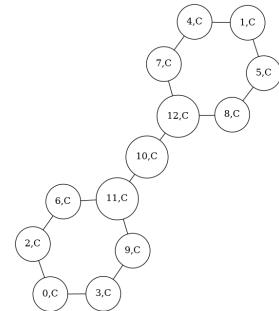
(a) Cluster 1



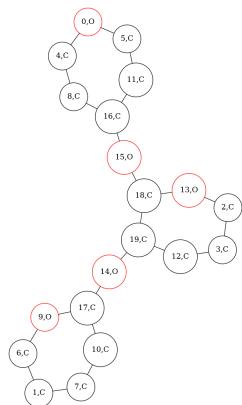
(b) Cluster 2



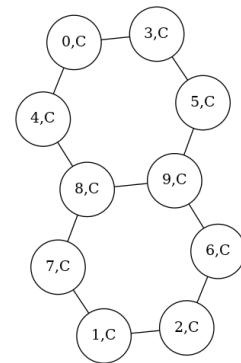
(c) Cluster 3



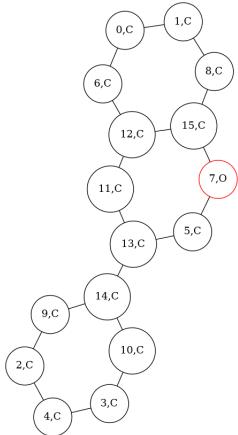
(d) Cluster 4



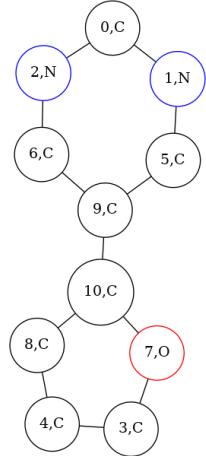
(e) Cluster 5



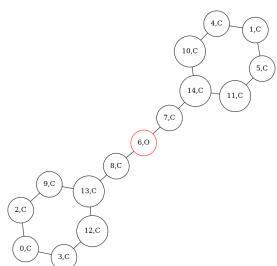
(f) Cluster 6



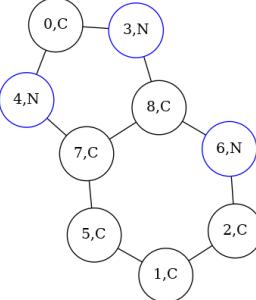
(g) Cluster 7



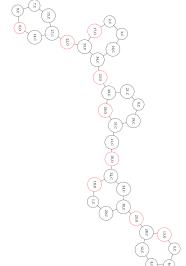
(h) Cluster 8



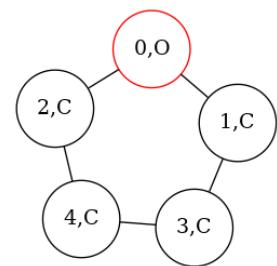
(i) Cluster 9



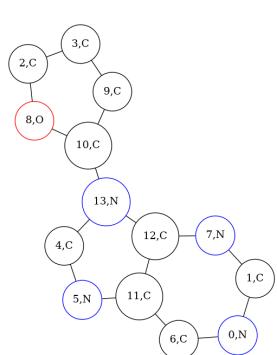
(j) Cluster 10



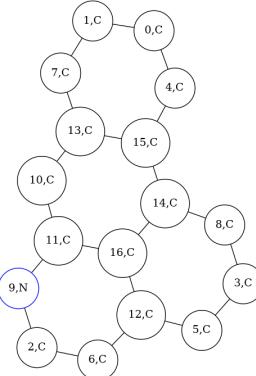
(k) Cluster 13



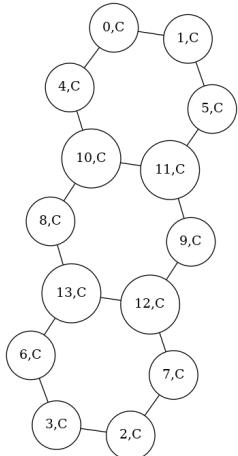
(l) Cluster 15



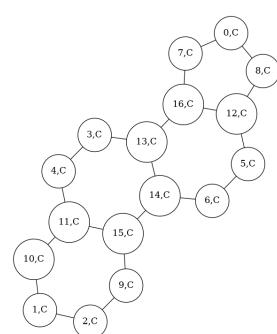
(m) Cluster 16



(n) Cluster 17



(o) Cluster 18



(p) Cluster 19

FIGURE 16 – Graphes moléculaires réduits d'un représentant de chaque cluster. Chaque atome est représenté par un sommet avec son ordre selon McKay ainsi que son symbole.

Certains clusters sont peu représentatifs d'un groupe de molécules similaires, c'est-à-dire qu'ils sont composés de molécules n'ayant pas forcément une structure proche. Il s'agit des clusters 11, 12 et 14. Ce résultat est visible sur la figure 15, puisque ces clusters ont une similarité d'intensité majoritairement bleue foncée.

Pour les autres clusters, nous trouvons au contraire une structure commune aux molécules qui les composent (voir figure 16).

Nous remarquons que les clusters 1, 2 et 3 sont représentés par des structures proches : un cycle de taille 6. Mais dans le cluster 1, le cycle est composé seulement de carbones, dans le cluster 2, de carbones et d'un autre atome (par exemple, un oxygène), et dans le cluster 3, de carbones et de plusieurs autres atomes. Cette différenciation est rendue possible par l'utilisation de la distance de Levenshtein.

Par comparaison, nous pouvons voir que le cluster 15 est composé des cycles de taille 5 sans discriminer sur le type des atomes. Le fait que tous les cycles de taille 5 ne forment qu'un seul cluster peut être dû à la sélection des molécules en amont. Il peut en effet y avoir peu de molécules avec un cycle de taille 5 parmi les 1000 sélectionnées.

Les clusters 4 et 9 parviennent à différencier la taille des chemins entre deux cycles de taille 6. Mais alors que dans le cluster 4 il y a toujours un atome entre les deux cycles, pour le cluster 9 le nombre d'atomes entre les cycles varient.

Le cluster 5 quant à lui regroupe les molécules ayant plusieurs cycles de taille 6 séparés par au moins 1 atome.

Ensuite certains clusters permettent de différencier les molécules qui possèdent plusieurs cycles partageant des atomes en commun. C'est le cas des clusters 6, 10, 17, 18 et 19. Le cluster 6 regroupe les molécules à deux cycles de taille 6, tandis que le cluster 10 représente les molécules à deux cycles de taille 5 et 6. De même, les clusters 17 et 19 rassemblent les molécules à 4 cycles, mais tous les cycles sont de taille 6 dans le premier, tandis qu'il y en a un de 5 atomes dans le deuxième. Le cluster 18 représente quant à lui les molécules à 3 cycles de taille 6.

Concernant les clusters 7 et 16, les structures sont similaires mais là encore une différence est faite entre les cycles de taille 5 et ceux de taille 6. Avec le cluster 8, nous retrouvons les molécules avec un cycle de taille 6 et un cycle de taille 5 séparés par une liaison.

Pour finir, nous avons le cluster 13, qui regroupe des molécules plutôt différentes, mais avec pour point commun que toutes les molécules sont des oses (monosaccharides) ou osides (oligosaccharides).

5 Perspectives

Dans le cadre de l'évolution de notre projet sur la comparaison des graphes de cycles moléculaires via la méthode MCIS, plusieurs axes d'amélioration et d'extension se présentent comme des opportunités prometteuses pour approfondir et enrichir notre recherche.

Premièrement, l'optimisation des algorithmes et structure de données notamment pour le MCIS s'avère cruciale, non seulement pour améliorer l'efficacité de nos analyses, mais aussi pour en accroître la précision. Cela pourrait inclure l'exploration de techniques de parallélisation ou l'adoption de nouvelles heuristiques plus efficaces. Une comparaison des résultats obtenus par le MCIS et par le MCES serait également pertinente, afin de s'assurer que notre approche cible au mieux les différences structurelles entre les molécules.

Un changement de structure de données serait aussi nécessaire pour la modélisation des graphes afin

de mieux prendre en compte la sparsité des liaisons dans les molécules. Les listes d'adjacence feraient ainsi un bon candidat de structure.

L'intégration de notre méthode avec d'autres techniques de comparaison moléculaire, comme les empreintes moléculaires ou les descripteurs chimiques, pourrait renforcer la robustesse et la profondeur de nos analyses comparatives. De plus, l'utilisation de l'apprentissage profond par réseaux de neurones pourrait être intéressant pour améliorer notre capacité à identifier et comparer automatiquement les sous-structures moléculaires complexes.

Pour finir, afin d'être sûr de l'intérêt de cette méthode, il serait intéressant de discuter avec un chimiste pour tester notre méthode sur des molécules précises et non aléatoires, et d'avoir un retour sur le bien-fondé de notre approche.

6 Conclusion

Pour conclure, ce projet a été une opportunité exceptionnelle de nous confronter à des défis de modélisation, d'algorithmique et d'implémentation, nous permettant de progresser considérablement dans notre compréhension et notre maîtrise de ces domaines. Nous pensons avoir mené ce projet à bien, en proposant une méthode d'analyse des molécules via leurs graphes de cycles, facilitant la formation de clusters de molécules similaires. Ce processus a inclus la récupération de molécules d'une base de données, leur sélection basée sur leur pertinence pour notre méthode, et l'application des algorithmes de McKay et Horton pour obtenir les graphes de cycles. La comparaison de ces graphes, en utilisant le MCIS et la similarité de Levenshtein, nous a permis de prendre en compte à la fois la structure moléculaire et la composition atomique dans nos calculs de similarité. L'implémentation d'une classification hiérarchique et la visualisation des résultats via une heatmap ont grandement facilité l'interprétation des données, révélant des patterns de similarité intrigants.

Ce projet nous a permis d'appliquer des connaissances apprises durant ce master. Nous avons également été confrontés à divers défis, notamment en termes d'organisation et de compréhension de concepts complexes. Ces difficultés ont souligné l'importance d'une gestion efficace du temps et d'une compréhension approfondie des travaux de recherche fondamentaux. Néanmoins, ces défis ont renforcé notre capacité à travailler en équipe, à distribuer les tâches efficacement et à apporter chacun nos connaissances et compétences uniques au projet. Cette expérience a également enrichi nos connaissances en algorithmique, en informatique.

En termes de perspectives, ce projet ouvre la voie à de nombreuses possibilités de recherche future. L'amélioration des algorithmes existants pour inclure les molécules sans cycles, l'exploration de nouvelles méthodes de calcul de similarité, et l'extension de notre travail à des bases de données moléculaires plus vastes sont autant de pistes à explorer.

Références

- [Blumenthal et al., 2020] Blumenthal, D. B., Boria, N., Gamper, J., Bougleux, S., and Brun, L. (2020). Comparing heuristics for graph edit distance computation. *The VLDB Journal*, 29(1) :419–458.
- [Bron and Kerbosch, 1973] Bron, C. and Kerbosch, J. (1973). Algorithm 457 : finding all cliques of an undirected graph. *Commun. ACM*, 16(9) :575–577.

- [Ehrlich and Rarey, 2011] Ehrlich, C. and Rarey, M. (2011). Maximum common subgraph isomorphism algorithms and their applications in molecular science : A review. *Interdisciplinary Reviews - Computation Molecular Science*, 1 :68–79.
- [Gruvaeus and Wainer, 1972] Gruvaeus, G. and Wainer, H. (1972). TWO ADDITIONS TO HIERARCHICAL CLUSTER ANALYSIS†. *British Journal of Mathematical and Statistical Psychology*, 25(2) :200–206.
- [Gu, 2022] Gu, Z. (2022). Complex heatmap visualization. *iMeta*, 1(3) :e43.
- [Gu et al., 2016] Gu, Z., Eils, R., and Schlesner, M. (2016). Complex heatmaps reveal patterns and correlations in multidimensional genomic data. *Bioinformatics*, 32(18) :2847–2849.
- [Hartke and Radcliffe, 2013] Hartke, S. and Radcliffe, A. (2013). McKay’s canonical graph labeling algorithm. *Contemporary Mathematics book series*, 479.
- [Hastings et al., 2016] Hastings, J., Owen, G., Dekker, A., Ennis, M., Kale, N., Muthukrishnan, V., Turner, S., Swainston, N., Mendes, P., and Steinbeck, C. (2016). Chebi in 2016 : Improved services and an expanding collection of metabolites. *Nucleic acids research*, 44(D1) :D1214—9.
- [Horton, 1987] Horton, J. D. (1987). A Polynomial-Time Algorithm to Find the Shortest Cycle Basis of a Graph. *SIAM Journal on Computing*, 16(2) :358–366.
- [Langfelder et al., 2008] Langfelder, P., Zhang, B., and Horvath, S. (2008). Defining clusters from a hierarchical cluster tree : the Dynamic Tree Cut package for R. *Bioinformatics*, 24(5) :719–720.
- [Levenshtein, 1966] Levenshtein, V. I. (1966). Binary Codes Capable of Correcting Deletions, Insertions and Reversals. *Soviet Physics Doklady*, 10 :707. ADS Bibcode : 1966SPhD...10..707L.
- [McKay and Piperno, 2014] McKay, B. D. and Piperno, A. (2014). Practical graph isomorphism, II. *Journal of Symbolic Computation*, 60 :94–112.
- [Murtagh and Legendre, 2014] Murtagh, F. and Legendre, P. (2014). Ward’s Hierarchical Agglomerative Clustering Method : Which Algorithms Implement Ward’s Criterion ? *Journal of Classification*, 31(3) :274–295.
- [Nouleho ilemo, 2020] Nouleho ilemo, S. (2020). *Algorithmique de graphes pour la similarité structurelle de molécules et de réactions*. PhD thesis, université Paris-Saclay.
- [Raymond, 2002] Raymond, J. W. (2002). RASCAL : Calculation of Graph Similarity using Maximum Common Edge Subgraphs. *The Computer Journal*, 45(6) :631–644.
- [Vismara, 1997] Vismara, P. (1997). Union of all the Minimum Cycle Bases of a Graph. *The Electronic Journal of Combinatorics*, 4(1) :R9.

Annexes

Annexe A Clusters

TABLE 1 – Tableaux représentant les ChEBI id des 1000 molécules comparées et leur appartenance à un cluster

Cluster	ChEBI id
1	35005, 84251, 83376, 28122, 73517, 147331, 73163, 88190, 86577, 78336, 19868, 19341, 62791, 75411, 18414, 91036, 55331, 67010, 195241, 60863, 45165, 133977, 46245, 53277, 132606, 89247, 90355, 2340, 131712, 83939, 18377, 58992, 35321, 27978, 83375, 133073, 9449, 8050, 70851, 64180, 57277, 77136, 33016, 143108, 49244, 74232, 1235, 42484, 74623, 20805, 61972, 84387, 131375, 86962, 86071, 136852, 115197, 28735, 53539, 87405, 60051, 28854, 32774, 89243, 195509, 68616, 83378, 53568, 28377, 28286, 145409, 65155, 64838, 137659, 86583, 32494, 29507, 86949, 53220, 59479, 61146, 38537, 143071, 84517, 53175, 182711, 53570, 28995, 190710, 18450, 20550, 17691, 23596, 18348, 190530, 4904, 132128, 142493, 85122, 64449, 58999, 48639, 45716, 83466, 71001, 61345, 93472, 36538, 31835, 64703, 75120, 86558, 73243, 583580, 27949, 131607, 62629, 88840, 64437, 46261, 33214
2	194110, 41105, 73230, 62346, 139286, 84768, 71098, 71110, 62168, 79145, 79132, 76297, 136926, 62321, 86510, 84160, 131396, 55499, 10295, 79149, 87075, 37557, 79152, 87043, 79154, 30701, 28518, 78970, 73781, 37715, 29051, 138511, 59364, 79049, 90515, 167500, 85497, 60206, 28137, 144708, 84170, 136952, 85474, 53028, 65275, 52884, 30620, 75965, 84721, 32167, 90098, 74003, 67149, 136875, 63762, 58807, 68583, 180671, 138007, 84756, 83892, 83878, 90514, 62495, 90342, 37679, 22798, 57874, 9162, 58658, 76239, 79311, 61185
3	78398, 29128, 26432, 133862, 30757, 33202, 38043, 28646, 26276, 27726, 30863
4	153729, 152314, 148368, 154730, 152991, 145598, 147563, 145615, 145470, 157223, 62353, 59314, 136857, 36226, 155419, 63592, 134034, 82989, 146900, 147144, 42528, 35430, 305790, 75915, 41967, 53216, 34579, 131938, 520819, 4640, 59613, 131184, 141775, 37923, 38345, 132964, 153538, 77100, 61741, 17531, 65018, 62435
5	73675, 16823, 90366, 53498, 60322, 53324, 27931, 134230, 156282, 84987, 63853, 148482, 148215, 133503, 146046, 68463, 59296, 88176, 87387, 35036, 5818, 30248, 90800, 62006, 61793, 137686, 62258, 58379, 90518, 139310, 88106, 142459, 61635, 65918, 140908, 31403, 79219, 62274, 138104, 86971, 63785, 61031, 147191, 148669, 48484, 18418, 68111, 38002, 81944, 18052, 6631, 71217, 64195, 90949, 145498, 39310, 39313, 144540, 62094, 60449, 64248, 64272, 31753, 142451, 193116, 83603
6	11173, 144078, 60922, 50645, 8361, 10319, 71996, 69206, 48446, 61873, 74804, 66661, 17763, 133565, 32155, 36119, 28012, 36625, 66030, 78887, 61186, 17001, 74688
7	70653, 66493, 5066, 28103, 67364, 142228, 85127, 79553, 75314, 144777, 75162, 3613, 65453, 17678, 71972, 2506, 55448, 9971, 140183, 55465, 65863
8	228294, 75179, 62904, 19780, 46398, 57502, 72778, 46960, 139232, 27244, 132191, 75078, 19068, 133764, 62086, 138241, 134690, 147384, 145865
9	138323, 78316, 28996, 50360, 143611, 31529, 32488, 133037, 60584, 180652, 149466, 83543, 84004, 83719, 81785, 50688, 83779, 133833, 139490, 145301, 16068, 193138, 58983, 53616, 49770, 77455, 6759, 61827, 90768, 136743, 156231, 41237, 69704, 63619, 136828, 139273, 92211, 91205, 79343, 77958, 59291, 81764, 38614, 17486, 16684
10	38476, 65054, 18426, 58209, 16083, 33070, 136516, 18299, 27056, 132892, 81938, 77027, 55466, 59951, 134105, 64139, 35586, 18375
11	38567, 38561, 38568, 67381, 69098, 140485, 75506, 75055, 47414, 78540, 71526, 65524, 66004, 109895, 66479, 28266, 137530, 51479, 17233, 79105, 79117, 1890, 138048, 66071, 36704, 22154, 65586, 139076, 5970, 50305, 15638, 1989, 84507, 72763, 132650, 177299, 63611, 39506, 145655, 141549, 52302, 52718, 74047, 28530, 16097, 73940, 73932, 65343, 65348, 64192, 71559, 63792, 58475, 28011, 15520, 189077, 15511, 140018, 140020, 140037, 140036, 41625, 64189, 58437, 62728, 3110, 58754, 27775, 83794
12	48949, 650657, 32922, 67136, 39381, 28529, 65860, 66773, 144439, 71511, 80944, 3737, 70201, 70204, 131831, 77588, 76574, 65356, 90686, 82844, 83942, 136895, 78847, 15774, 16846, 68518, 701, 47781, 83532, 66549, 3996, 38887, 60076, 63107, 91005, 167489, 81960, 22151, 180490, 180472, 64193, 52226, 10216, 76804, 64901, 528918, 86489, 66721, 139168, 73452, 80214, 32446, 85387, 59358, 139229, 68421, 66742, 62403, 73041, 68166, 167159, 131596, 16081, 62384, 140909, 52134, 31615, 31520, 40, 86635, 131821, 32649, 3133, 91460, 87245, 37831, 66876, 77925, 78549, 63663, 53669, 57482, 10023, 83168, 31622, 4858, 92511, 73274, 88328, 90842, 7495, 3744, 145791, 90911, 68340, 139128, 60397, 60733, 90541, 44032, 60998, 139243, 75361, 70751, 172918, 152560, 102131, 140233, 77908, 36427, 90929, 191396, 207229, 68246, 36325, 64287, 90323, 66118, 58018, 67468, 192379, 65930, 78577, 67310, 51756, 65666, 65885, 34993, 39228, 9044, 66313, 166972, 15738, 9984, 66337, 142834, 70404, 183634, 34704, 652822, 132634, 66401, 52432, 83819, 28271, 282234, 65831, 63514, 152056, 52167, 66516, 138475, 66438, 69221, 69509, 156301, 28051, 37777, 35652

Continue sur la page suivante

Table 1 : Tableaux représentant les ChEBI id des 1000 molécules comparées et leur appartenance à un cluster (suite)

Cluster	ChEBI id
13	147243, 61522, 62119, 156281, 61315, 71595, 71042, 28278, 144089, 84671, 142444, 150468, 88088, 74976, 59215, 59228, 59626, 87369, 59636, 189582, 63807, 77451, 60625, 71471, 64048, 83548, 150940, 142462, 132516, 70965, 134678, 149615, 87699, 87621, 53019, 151211, 173087, 90385
14	146650, 63144, 75621, 75643, 192979, 80716, 46416, 66646, 51401, 87468, 88280, 76842, 76058, 69604, 2938, 82822, 33120, 37415, 51390, 58323, 62799, 65363, 136457, 144361, 172908, 84463, 140996, 52563, 65961, 33014, 64376, 51000, 36523, 32399, 140175, 140171, 76701, 75695, 16089, 82800, 82799, 38257, 57307, 36159, 52558, 51218, 52488, 60064, 28115, 27786, 90967, 64668, 15433, 60344, 83282, 36320, 36305, 57437, 85193, 134432, 164196, 73696, 53236, 176852, 66587, 68864, 143255, 66260, 52724, 66954, 52158, 51449, 156449, 90801, 55403, 80788, 26590, 50040, 51318, 30270, 37145, 57465, 87991, 131398, 18053, 39102, 136988, 138262, 133935, 138613, 138263, 75700, 131974, 133938
15	62838, 45636, 87572, 63186, 32518, 76620, 32523, 63404, 73520, 74745, 74661, 73383, 35561, 74761, 33137, 16373, 82725, 31132, 73337, 62912, 50992, 57862, 76248, 76247, 71161, 74159, 75592, 62877, 76037, 61560, 18274, 78346, 133743, 83834, 53446, 138315, 133185, 88956, 18413, 30847, 148031, 145606, 189588, 28272, 83435, 87296, 39184, 41254, 145738, 58694, 40595, 48570, 4250, 37736, 57399, 63976, 139085, 48695, 111510, 28045, 57406, 140267, 59427
16	17980, 65330, 67009, 62034, 74277, 131551, 57673, 76484, 15414, 37554, 15496, 74587, 74972, 60043, 77334, 84312, 33440, 16761, 86376, 27402, 74086, 57339, 76389, 61123, 86042, 136758, 76360, 64654, 76367, 90817, 22263, 75068, 74109, 62047, 76687, 74445, 52966, 37666, 57351, 12060, 84650
17	8691, 65529, 66159, 136557, 87749, 35643, 32216, 59071, 70562, 66075, 9746, 81347, 37450, 5730, 72440, 90220, 52163, 66252, 132718, 51825, 254496, 91276, 67907, 68369, 69110, 137799, 136886, 68591, 41633, 71418, 67289, 71025, 16512, 141633, 68042, 69503, 69502
18	59536, 63314, 9216, 132306, 53771, 167214, 138224, 66662, 65607, 50931, 67141, 51055, 50756, 45980, 50064, 37456, 70569, 40753, 36547, 70566, 64450, 33835, 138968, 71653
19	166888, 60008, 84352, 31184, 62228, 71542, 8389, 71187, 134124, 52386, 156480, 35419, 16962, 182127, 52322, 88102, 16973, 68634, 137688, 194032, 20538, 63824, 88108, 35512, 16784, 34477, 16581, 81471, 78699, 137053, 17639, 70117, 70110, 62044, 190404, 87731