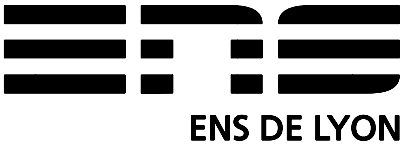

Licence Informatique Fondamentale
École Normale Supérieure de Lyon
Université Claude Bernard Lyon I

Stage 2015–2016
Ruben Staub
L3 Informatique Fondamentale

Calculs d'isomorphismes pour la Chemoinformatique

Résumé : *Le développement de cages moléculaires représente de nombreuses applications chimiques, et les progrès récents dans ce domaine entraîne le besoin d'outils performants permettant la génération exhaustive des cages réalisables à partir un ensemble de blocs de base chimiques prédéfini. Pour cela, une méthode algorithmique utilisant une modélisation justifiée basée sur la théorie des graphe a été développée au laboratoire DAVID. Cependant, afin d'augmenter les performances de cette méthode, il est nécessaire de pouvoir calculer efficacement les classes d'isomorphismes d'un ensemble de plongements sphériques de graphes planaires orientés dont les sommets et arcs sont colorés. Le but de ce stage étant l'optimisation de ce partitionnement, différentes voies d'améliorations ont été étudiées et implémentées, et une méthode alternative a été proposée, implémentée et testée.*

Mots clefs : *Calcul d'isomorphismes ; Cages moléculaires ; Chemoinformatique ; C*

Stage encadré par : Yann Strozecki
yann.strozecki@uvsq.fr / tél. (+33) 1 39 25 43 12
Laboratoire DAVID
Université de Versailles Saint-Quentin en Yvelines
45 Avenue des États-Unis
78035 Versailles
http://www.david.uvsq.fr/

28 août 2016

Table des matières

1 Problématique

La synthèse chimique consiste en la construction de structures chimiques à partir de «briques élémentaires» (habituellement, des molécules faciles d'accès et/ou peu chères).

Les avancées en synthèse chimique ont permis de synthétiser des composés chimiques de plus en plus grands et complexes. On s'intéressera ici à la synthèse de cages moléculaires.

1.1 Exposition du problème chimique

Une cage moléculaire consiste en un assemblement d'atomes liés par des liaisons chimiques, et formant une cage tridimensionnelle. En réalité, la synthèse chimique de ces cages ne se fait pas atome par atome, mais par un assemblage de molécules (un assemblement d'atomes liés par liaisons chimiques, représentant une «brique élémentaire») pouvant réagir entre elles, et ainsi former de nouvelles liaisons entre ces unités chimiques modulaires. Ceci est particulièrement vrai pour la «Chimie clic»[10] qui s'appuie sur l'assemblement spontané de motifs chimiques modulaires, inspiré des synthèses biochimiques dans la nature.

Le développement de cages moléculaires représente de nombreuses applications allant de l'industrie pharmaceutique à la catalyse.

En effet, une cage moléculaire peut être conçue afin de recueillir une autre molécule, appelée substrat, en formant des liaisons chimiques avec ce dernier, impliquant des sites actifs répartis sur la face intérieure. Ainsi, le substrat peut être soumis à une modification de sa réactivité et y subir une réaction chimique (catalyse) ou non (transport de principes actifs médicamenteux, ...).

De plus, une cage moléculaire peut être conçue pour interagir de façon contrôlée avec son environnement, au travers de sa taille, ou des propriétés chimiques de sa face extérieure.

Remarquons que pour ces deux caractéristiques, il est primordial de pouvoir contrôler la taille d'une cage moléculaire, et de pouvoir définir une face extérieure et une face intérieure.

Nous ne nous intéresserons pas au détail de la synthèse, pour nous concentrer sur la combinatoire des cages moléculaires réalisables en partant d'un ensemble prédéfini de motifs chimiques de base.

Cette approche permet ainsi d'explorer les produits éventuels d'une synthèse avant de la réaliser. Permettant ainsi d'économiser le coût d'une synthèse finalement peu intéressante et/ou de considérer de nouvelles cages qui n'avaient pas été imaginées.

1.2 Modélisation

Afin de produire une génération exhaustive des cages moléculaires possibles d'une taille donnée à partir d'un ensemble de motifs chimiques prédéfinis, la modélisation adoptée[4] sera détaillée dans cette section.

1.2.1 Définitions

Graphe : *Un graphe* $\mathrm{G} = (\mathrm{V}, \mathrm{E})$ *est défini par la donnée d'un multiensemble* V *de sommets (ou nœuds), et d'un multiensemble* E *d'arêtes, chaque arête étant définie par une paire non ordonnée de sommets de* V.

Le degré d'un sommet u *est défini comme* $|\{\!\{\{\!\{u, v\}\!\} \in \mathrm{E} \ ; \ v \in \mathrm{V}\}\!\} \cup \{\!\{\{\!\{u, u\}\!\} \in \mathrm{E}\}\!\}|$ [1].

Une boucle est une arête de la forme (u, u) *avec* $u \in \mathrm{V}$.

1. la notation $\{\!\{\ldots\}\!\}$ dénote un multiensemble

Graphe orienté : *Un graphe orienté* $\mathrm{G} = (\mathrm{V}, \mathrm{E})$ *est défini par la donnée d'un multiensemble* V *de sommets, et d'un multiensemble* E *d'arcs, chaque arc étant défini par un couple (une paire ordonnée) de sommets de* V.

Un arc e est dit sortant pour un sommet u, s'il est de la forme $e = (u, v),\ v \in \mathrm{V}$, *et entrant s'il est de la forme* $e = (v, u)$.

Le degré d'un sommet est alors la somme du semi-degré entrant (défini par $|\{\!\{(u, v) \in \mathrm{E}\ ;\ v \in \mathrm{V}\}\!\}|$*) et du semi-degré sortant (défini par* $|\{\!\{(v, u) \in \mathrm{E}\ ;\ v \in \mathrm{V}\}\!\}|$*).*

Multigraphe : *Un graphe* $\mathrm{G} = (\mathrm{V}, \mathrm{E})$ *est dit multigraphe si on autorise :*

- *les arêtes (ou arcs) de la forme* (u, u) *avec* $\mathrm{u} \in \mathrm{V}$, *appelées boucles.*
- *plusieurs arêtes (ou arcs) de* E *ayant le même couple de sommets de* V, *appelées arêtes (ou arcs) multiples.*

Connexité : *Soit u et v des sommets disjoints de* V. *Une chaîne d'arcs reliant u à v est une suite d'arcs* $((a_n, b_n))_{n \in \{0,\ \ldots,\ \mathrm{N}\}}$ *avec* $\mathrm{N} \in \mathbb{N}$ *telle que :*

- $a_0 = u$
- $b_\mathrm{N} = v$
- $\forall i \in \{0,\ \ldots,\ \mathrm{N} - 1\}, \quad b_i = a_{i+1}$

Un graphe orienté $\mathrm{G} = (\mathrm{V}, \mathrm{E})$ *est dit connexe si pour tout couple de sommets disjoints* (u, v) *de* V^2, *il existe une chaîne d'arcs de* E *reliant u à v.*

Planarité : *Une représentation sphérique d'un graphe orienté* $\mathrm{G} = (\mathrm{V}, \mathrm{E})$ *est la donnée :*

- *d'un ensemble de points disjoints deux à deux, sur une sphère préalablement fixée, correspondant aux sommets de* V.
- *d'un ensemble de courbes tel que pour tout arc de* E *défini par deux sommets de* V *il existe une unique courbe sur la sphère, dans cet ensemble, qui relie les deux points correspondants ; et tel qu'il n'existe pas deux courbes qui se croisent.*

Un graphe orienté est dit planaire s'il admet une représentation sphérique[2].

Plongement : *Un plongement sphérique (ou carte) d'un graphe* G *est l'information topologique d'une représentation sphérique de* G. *En pratique, cette information topologique supplémentaire se traduit par la donnée d'un ordre (à permutation circulaire près) sur les arêtes (ou arcs) adjacents à un même nœud.*

Nous modéliserons alors une cage moléculaire $\mathcal{C}$ par un plongement sphérique d'un multigraphe orienté connexe planaire (muni d'une coloration des sommets et des arcs) noté $\mathcal{M}_\mathcal{C}$ et abrégé sous le terme de carte.

À partir de ces définitions, on va pouvoir justifier la modélisation adoptée pour représenter une cage moléculaire et ses caractéristiques chimiques.

1.2.2 Modélisation des motifs chimiques :

Chaque motif chimique est représenté par un nœud avec une couleur associée.

En effet, du point de vue combinatoire, il est plus naturel de considérer une cage moléculaire dans sa globalité comme un assemblage de motifs, au lieu d'étudier l'assemblage d'atomes sous-jacent (qui relèverait plutôt du travail de synthèse chimique).

2. Plus précisément, une représentation planaire. Mais une sphère étant topologiquement équivalente à un plan privé d'un point, et les graphes considérés étant finis, les deux représentations sont homéomorphes. La définition est ainsi correcte.

De plus, cette représentation permet de s'abstraire de l'éventuelle altération des motifs liée au processus réactionnel.

Ces motifs, représentés par des nœuds, peuvent interagir les uns avec les autres au travers de liaisons chimiques, dont nous allons détailler la représentation.

1.2.3 Modélisations des liaisons chimiques :

Chaque liaison chimique entre deux motifs met en jeu un site actif de chacun des motifs, appelé groupe (ou fonction) caractéristique. Ainsi, chaque groupe caractéristique d'un motif sera représenté par un arc sortant du nœud associé à ce motif (pour l'instant, on ne considère pas son extrémité non reliée).

Il est nécessaire de pouvoir accéder à la nature d'un groupe caractéristique, puisque chaque type de groupe caractéristique possède sa propre réactivité. Par conséquent, une couleur est attribuée à chaque arc correspondant à la nature du groupe caractéristique associé.

À chaque couleur d'arc est associée une couleur complémentaire unique, telle qu'une liaison chimique ne puisse se faire qu'entre deux groupes caractéristiques dont les couleurs associées sont complémentaires l'une de l'autre.

Remarque : On considère ici que chaque type de groupe caractéristique ne peut réagir qu'avec un seul autre type de groupe caractéristique, prédéfini. En effet, on peut se ramener à ce cas en dupliquant les motifs dont les groupes caractéristiques sont peu sélectifs. De plus, une réactivité peu sélective se traduit souvent en pratique par de mauvais rendements lors de la synthèse expérimentale, du fait de la présence de produits parasites, ce qui limite leur utilisation.

Par conséquent, une liaison chimique entre un motif A et un motif B se traduit dans ce modèle par une paire d'arcs (A, B) et (B, A) avec des couleurs complémentaires correspondant respectivement à la nature du groupe caractéristique impliqué dans le motif A et le motif B.

Remarque : Seules les liaisons à deux centres sont considérées, puisque les liaisons à plusieurs centres[5][6] peuvent toujours être modélisés par des liaisons à deux centres. [3]

1.2.4 Modélisation d'une cage planaire

Les multiensembles de nœuds $V_{\mathcal{C}}$ colorés et d'arcs $E_{\mathcal{C}}$ colorés permettent ainsi de modéliser une cage moléculaire $\mathcal{C}$ sous la forme d'un graphe orienté $G_{\mathcal{C}} = (V_{\mathcal{C}}, E_{\mathcal{C}})$. Nous allons justifier plus en détails les spécificités de ce graphe.

Multigraphe : Une liaison pouvant se faire entre deux groupes caractéristiques d'un même motif, $G_{\mathcal{C}}$ peut contenir des boucles. De plus, deux motifs peuvent être liés par plusieurs liaisons chimiques. Ainsi, $G_{\mathcal{C}}$ peut contenir des arcs multiples.

Par conséquent, $G_{\mathcal{C}}$ est un multigraphe dans le cas général.

Connexité : Une cage moléculaire $\mathcal{C}$ est une structure qui ne peut pas être composée de plusieurs assemblages non liés entre eux (une telle structure ne pourrait pas conserver son intégrité en pratique).

Ainsi, le multigraphe orienté $G_{\mathcal{C}}$ associé est connexe.

Plongement sphérique : On considérera les cages moléculaires pouvant être vues comme une surface monocouche homéomorphe [4] à une sphère [5]. Ainsi, on peut considérer que les liaisons chimiques d'intérêt s'établissent sur une sphère.

3. Quitte à considérer les interactions entres les motifs au lieu des réactions au sens strict.

4. Deux surfaces sont dites homéomorphes s'il existe une bijection continue de réciproque continue allant de l'une vers l'autre (autrement dit, elles ont la même topologie).

5. Quitte à redéfinir les motifs, ou à décomposer la cage multicouche en monocouches.

Or, deux liaisons chimiques ne pouvant pas se croiser, on modélise une cage moléculaire $\mathcal{C}$ par un unique plongement sphérique $\mathcal{M}_\mathcal{C}$ pour le multigraphe orienté connexe planaire $G_\mathcal{C}$ associé. Cette information topologique supplémentaire se traduit par la donnée d'un ordre (à permutation circulaire près) dans le parcours des arcs autour d'un nœud.

Inclure exemple 2 plongements possibles

Orientation des motifs : Cet ordre sur les arcs autour d'un nœud est ici prédéfini pour chaque type de motif.

En effet, la disposition des groupes caractéristiques autour d'un motif est fixée, et ne dépend que de la nature du motif.

Remarque : Cet ordre (à permutation circulaire près) des groupes caractéristiques autour d'un motif implique que l'on distingue un motif de son image dans un miroir (ou juste le même motif «retourné» selon un axe qui est pas un axe de symétrie pour cette molécule), ce qui peut sembler plus limitant, puisque les motifs se présentent sous toutes les orientations possibles dans le milieu réactionnel.

Cependant, cela permet de choisir l'orientation des motifs utilisés, afin de sélectionner par exemple quelle face d'un motif sera à l'extérieur de la cage.

De plus, il est toujours possible d'autoriser les retournements en dupliquant les motifs initiaux. Et enfin, cela permet de faire la distinction entre des composés chiraux [6] entre eux.

Saturation des sites actifs : Soit $\mathcal{C}$ une cage moléculaire. On dira que le multigraphe orienté connexe planaire $G_\mathcal{C}$ (ou le plongement sphérique $\mathcal{M}_\mathcal{C}$ pour $G_\mathcal{C}$) associé est saturé si tous les groupes caractéristiques de $\mathcal{C}$ sont impliqués dans des liaisons chimiques. On dira que $\mathcal{M}_\mathcal{C}$ et $G_\mathcal{C}$ sont insaturés sinon.

1.3 But

1.3.1 Partitionnement en classes d'isomorphismes

Le but du projet est la génération exhaustive des cages moléculaires réalisables en partant d'un ensemble prédéfini de motifs chimiques de base. Dans le cadre de la modélisation adoptée, cela revient à répertorier les classes d'isomorphismes des cartes (ou plongements sphériques) générés exhaustivement à partir d'un ensemble prédéfini de motifs chimiques de base.

Dans le cadre de ce stage, nous ne nous intéresserons pas à la génération exhaustive du multiensemble $\mathcal{S}$ des cartes associé à un ensemble prédéfini de motifs chimiques de base.

Nous nous restreindrons plutôt à la classification de $\mathcal{S}$ en classes d'isomorphismes, et en particulier le but initial du stage est l'optimisation de cette étape (à partir d'une approche déjà implémentée en C par le laboratoire). En effet, la génération de $\mathcal{S}$ peut fournir, dans son implémentation actuelle, des copies isomorphes qui correspondent ainsi à une même cage moléculaire.

Ce partitionnement [7] permet d'éliminer les copies inutiles, afin de réduire le nombre de cartes dont la stabilité et l'intérêt doivent être estimés [8] pour le chimiste. Cette estimation étant particulièrement coûteuse, le surcoût lié à l'élimination des solutions isomorphes sera largement compensé par le gain obtenu lors de cette estimation.

6. Un composé chiral est une molécule telle que son image dans un miroir n'est pas superposable avec elle-même.
7. Cette approche peut s'adapter à la gestion des bases de données devant être classifiées en classes d'isomorphismes.
8. Les critères choisis ici ne feront pas l'objet de cet étude.

1.3.2 Performance algorithmique

Le nombre de cartes à tester (pour la relation d'équivalence associée aux isomorphismes) est un problème combinatoire complexe[3], pouvant être exponentiel en la taille donnée en entrée pour chaque solution.

Par conséquent, les instances étudiées pour le calcul d'isomorphisme peuvent raisonnablement être considérées comme de taille moyenne. En effet, l'explosion combinatoire du nombre de solutions domine souvent l'accroissement de la taille des solutions elles-mêmes, pour ce qui est du temps de calcul total. Autrement dit, si des instances de grandes tailles apparaissent, le nombre total d'instances à calculer est souvent déjà trop grand pour que le calcul aboutisse dans des temps raisonnables.

S'il est ainsi peu nécessaire de s'intéresser au comportement asymptotique des algorithmes choisis, il est par contre primordial de privilégier les algorithmes performants pour des instances de taille moyenne à petite.

2 Contexte

Le problème de l'isomorphisme de graphes est un problème classique, possédant de nombreuses applications allant du design de circuits imprimés[2] à la reconnaissance d'empreintes digitales[13]. Particulièrement utilisé en Chemoinformatique, il s'applique à la fois à la gestion de bases de données et à la recherche et l'indexation de molécules (via la définition identifiants uniques : SMILES[16], InChi[7], ...).

2.1 Isomorphismes en général

Définition : *Soit* $\mathrm{G}_1 = (\mathrm{V}_1, \mathrm{E}_1)$ *et* $\mathrm{G}_2 = (\mathrm{V}_2, \mathrm{E}_2)$ *deux graphes. Un isomorphisme de graphe de* G_1 *dans* G_2 *est une bijection* ϕ *de* V_1 *dans* V_2 *qui vérifie :*

$$\forall (i,j) \in \mathrm{V}_1^2, \quad (i,j) \in \mathrm{E}_1 \Leftrightarrow (\phi(i), \phi(j)) \in \mathrm{E}_2$$

On dit que G_1 *et* G_2 *sont isomorphes s'il existe un isomorphisme de graphe de* G_1 *dans* G_2*.*

Théorème : *Décider si deux graphes sont isomorphes est un problème appartenant à la classe NP*[9]*.*

Preuve : Soit $\mathrm{G}_1 = (\mathrm{V}_1, \mathrm{E}_1)$ et $\mathrm{G}_2 = (\mathrm{V}_2, \mathrm{E}_2)$ une instance du problème. On considère ϕ, l'isomorphisme de graphe de G_1 dans G_2 comme certificat. On peut alors vérifier en $O(\mathrm{E}_1)$ que ϕ est bien un isomorphisme de graphe de G_1 dans G_2, en testant $\{\!\{(\phi(u), \phi(v)) \,;\, (u,v) \in \mathrm{E}_1\}\!\} \stackrel{?}{=} \mathrm{E}_2$.[10]

Cependant, il n'existe actuellement pas de preuve de la NP-complétude de ce problème. De même, il n'existe pas d'algorithme polynomial pour résoudre ce problème de décision. La meilleure complexité algorithmique en temps obtenue à ce jour pour ce problème est quasi-polynomiale en $\exp(\log(n)^{O(1)})$[1].

2.2 Cas particulier des graphes planaires

Toutefois, pour certaines classes de graphes le problème est plus simple, et peut se résoudre en temps polynomial. C'est le cas en particulier pour les arbres[11], les graphes de degré borné[11], les graphes planaires[15], ainsi que les cartes, sur lesquels nous nous focaliserons ici.

9. La classe NP est formée des problèmes de décision qui possèdent un vérifieur polynomial.
10. la notation $\{\!\{\ldots\}\!\}$ est nécessaire ici pour le cas général des multigraphes où E_1 et E_2 sont alors des multiensembles.
11. Un arbre est un graphe non orienté, connexe, acyclique.

La section suivante exposera quelques algorithmes fondateurs de complexité polynomiale en temps, prouvant l'appartenance du problème étudié à la classe P [12].

2.3 Résumé bibliographique

Étudions les méthodes de résolution classiques à travers deux approches générales de référence.

2.3.1 Réduction

Une première approche[8] consiste à appliquer des réductions successives sur deux cartes, simultanément, jusqu'à obtenir :

- Soit une impossibilité d'appliquer la même réduction aux deux cartes, signifiant alors l'absence d'isomorphisme entre ces deux cartes.
- Soit des cartes qui ne sont plus réductibles, et qu'il suffit de comparer. Or, l'ensemble des cartes irréductibles étant fini, cette comparaison finale est bornée en temps par une constante.

Les réductions doivent ainsi satisfaire certaines propriétés :

- Conservation de l'information, afin que les cartes après réductions soient isomorphes si et seulement si les cartes avant réduction le sont.
- Commutativité des transformations, pour un même type de réduction. Cela assure une indépendance du résultat par rapport à l'ordre des réductions de même type appliquées.

Des labels sont fixés initialement pour les sommets et les arêtes.

Les différents types de réductions sont alors appliquées selon un ordre de priorité prédéfini. Par conséquent, l'application de toutes les réductions dont la priorité est supérieure à une certaine valeur, aboutit à une forme canonique. En effet, soit les réductions sont commutatives, soit l'ordre dans lequel elles sont appliquées est prédéfini par l'ordre de priorité associé.

Cet approche permet d'obtenir une complexité en temps linéaire par rapport à la taille des graphes en entrée[8]. Cependant, cette méthode n'est pas destinée à être implémentée en pratique, en particulier pour les instances de petite et moyenne taille.

2.3.2 Partitionnement

Une autre approche utilisant un partitionnement par rapport à une valeur est également décrite dans la littérature[9], et résumée ici.

Soit $\mathcal{M}_1 = (\mathrm{V}_1, \mathrm{E}_1)$ et $\mathcal{M}_2 = (\mathrm{V}_2, \mathrm{E}_2)$ deux plongements sphériques de graphes orientés (ou cartes), dont l'information topologique supplémentaire est stockée sous la forme d'un ordre des arcs (entrants et sortants) autour de chaque sommet.

La valeur de discrimination est définie par une application λ de $\mathrm{E}_1 \cup \mathrm{E}_2$ dans $\mathbb{N}$, telle que $\lambda(e_1) = \lambda(e_2)$ si et seulement si :

- La taille de la face [13] située à droite/gauche (définie à une convention fixée près) de l'arc e_1 est égale à la taille de la face située à droite/gauche de e_2.
- Soit $e_1 = (u_1, v_1)$ et $e_2 = (u_2, v_2)$. Le degré de u_1/u_2 est le même que le degré de v_1/v_2.

On définit également un chemin primaire comme une chaîne d'arcs $((u_n, u_{n+1}))_{n \in \{0,\ \ldots,\ \mathrm{N}\}}$ avec $\mathrm{N} \in \mathbb{N}$ telle que pour $0 < i \leq N$, (u_i, u_{i+1}) est l'arc suivant ou précédant l'arc (u_{i-1}, u_i) autour du sommet u_i, à permutation circulaire près (selon l'ordre donné par le plongement sphérique).

12. la classe P est formée des problèmes pouvant être décidés en temps polynomial en la taille de l'entrée.

13. Une face est une chaîne d'arcs dont le plongement sphérique délimite une surface connexe de la sphère telle que son intérieur (i.e. privée de son bord) n'intersecte aucune courbe du plongement.

Deux chemins sont dits correspondants si la même procédure de choix déterministe [14] est appliquée. Ainsi, deux chemins principaux sont correspondants lorsque la sélection entre l'arc précédant/suivant est la même dans les deux cas.

Enfin, on définit deux arcs e_1 et e_2 comme étant distinguables, s'il existe des arcs e_3 et e_4, un chemin primaire de e_1 à e_3 et un chemin primaire correspondant de e_2 à e_4, tels que $\lambda(e_3) \neq \lambda(e_4)$. Dans le cas contraire, on dit que e_1 et e_2 sont indistinguables.

L'algorithme étudié s'appuie sur le fait que pour tout qu'il existe un isomorphisme ϕ de $\mathcal{M}_1$ dans $\mathcal{M}_2$ tel que $\phi(e_1) = e_2$ avec $e_1 \in \mathrm{E}_1$ et $e_2 \in \mathrm{E}_2$ si et seulement si e_1 et e_2 sont indistinguables[9].

Ainsi, les arêtes sont initialement regroupés en blocs selon leur valeur par λ. Puis les blocs sont partitionnés successivement selon leur voisinage avec les arêtes des autres blocs, de telle sorte qu'à la fin du processus, deux arêtes sont dans un même bloc si et seulement s'ils sont indistinguables.

Le partitionnement successif utilisé permet d'arriver à une complexité en temps en $O(|\mathrm{E}|\log|\mathrm{E}|)$ où $\mathrm{E} = \mathrm{E}_1 \cup \mathrm{E}_2$.

2.4 Approche initiale du laboratoire

En pratique, pour la comparaison des cages moléculaires, les cartes associées sont de taille relativement petite. Ainsi, une approche alternative a été utilisée, exploitant les caractéristiques des cartes en entrée.

2.4.1 Principe général

L'algorithme repose sur la conversion canonique d'une carte en un mot, appelé signature, qui contient suffisamment d'information pour reconstruire la carte associée.

Soit σ la fonction qui à une carte $\mathcal{M}$ associe une signature canonique $\sigma(\mathcal{M})$. Pour comparer deux cartes $\mathcal{M}_1$ et $\mathcal{M}_2$ il suffit alors de comparer $\sigma(\mathcal{M}_1)$ et $\sigma(\mathcal{M}_2)$.

Ainsi, pour partitionner l'ensemble $\mathcal{S}$ en classes d'isomorphismes, la signature de chaque élément $\mathcal{M} \in \mathcal{S}$ est calculée et comparée aux signatures représentant les classes d'isomorphismes déjà trouvées.

Cette approche repose alors fortement sur la définition de la signature utilisée, que nous allons étudier plus en détail.

2.4.2 Signature

Définition : *On définit comme signature (ou signature forte) de la carte $\mathcal{M}$ le mot $\sigma(\mathcal{M})$, tel que σ est une injection, c'est-à-dire vérifiant la propriété :*

$$\forall \mathcal{M}_1 \forall \mathcal{M}_2, \quad \mathcal{M}_1 = \mathcal{M}_2 \Leftrightarrow \sigma(\mathcal{M}_1) = \sigma(\mathcal{M}_2), \quad \text{avec } \mathcal{M}_1 \text{ et } \mathcal{M}_2 \text{ des cartes.}$$

Par la suite, on associera également le terme de fonction de signature à l'application σ, pour plus de lisibilité.

On définit comme signature faible de la carte $\mathcal{M}$ un mot $\sigma'(\mathcal{M})$, tel que :

$$\forall \mathcal{M}_1 \forall \mathcal{M}_2, \quad \mathcal{M}_1 = \mathcal{M}_2 \Leftarrow \sigma'(\mathcal{M}_1) = \sigma'(\mathcal{M}_2), \quad \text{avec } \mathcal{M}_1 \text{ et } \mathcal{M}_2 \text{ des cartes.}$$

14. à l'étape i, (u_i, u_{i+1}) est choisi comme le $k_i$ème arc suivant l'arc (u_{i-1}, u_i) dans l'ordre des arcs autour du sommet u_i. Une procédure de choix est alors la donnée de la suite $(k_i)_i$.

Remarquons alors que si un mot $\sigma'(\mathcal{M})$ (généré à partir d'une carte $\mathcal{M}$) permet à lui-seul de reconstruire la carte $\mathcal{M}$, alors il s'agit d'une signature faible de $\mathcal{M}$.

Une fonction de signature σ doit alors posséder les propriétés suivantes :
- Unicité de l'image : à une carte $\mathcal{M}$ on associe une unique signature $\sigma(\mathcal{M})$.
- Injectivité : conservation de l'information, telle que pour toute carte $\mathcal{M}$, il suffit d'avoir $\sigma(\mathcal{M})$ pour reconstruire $\mathcal{M}$ (i.e. bijectivité de σ sur son image).

Une «bonne» fonction de signature σ est définie par les critères suivants :
- Rapidité du calcul de l'image de $\mathcal{S}$ par σ.
- Rapidité de la comparaison entre deux signatures.

Les parcours de graphes étant habituellement capables de couvrir rapidement le graphe, de manière assez canonique, il est alors commun d'y avoir recours pour extraire une information globale sur un graphe.

La signature σ_0 choisie initialement est ainsi construite à partir d'un parcours de graphe.

Définition : On définit notre parcours DFS modifié par le pseudo-code suivant, $\mathcal{M} = (\mathrm{V}, \mathrm{E})$ [15] étant une carte saturée (plongement sphérique d'un multigraphe orienté planaire saturé), $u \in \mathrm{V}$ étant un sommet de $\mathcal{M}$, et $e = (u, v),\ v \in \mathrm{V}$ un arc de $\mathcal{M}$:

```
1  DFS_modif(C,u,e):
2    sig ← ∅;
3    pour u dans V:
4      rang[u] ← -1;
5    tour ← 0;
6    ajouter (u,e) dans pile P;
7    ajouter comp(u,e) dans pile P;
8    tant que P est non vide:
9      retirer dernier objet de P et le stocker dans (u,e);
10     si rang[u] = -1:
11       rang[u] ← tour;
12       e' ← suiv(u,e);
13       tant que e' ≠ e:
14         ajouter comp(u,e') dans P;
15         e' ← suiv(u,e');
16     sig ← ajout_sig(sig,u,e);
17     tour++;
18   retourne sig;
```

avec :
- `comp(u,e)` étant le couple (v, e^c) où e est un arc reliant u à v, et e^c est l'arc (reliant v à u) complémentaire de e, tel que la paire (e, e^c) représente une liaison chimique (voir section 1.2.3).
- `suiv(u,e)` étant l'arc suivant e dans l'ordre des arcs sortants autour de u (à permutation circulaire près).
- `sig ← ajout_sig(sig,u,e)` rajoute (concatène) à la signature faible *sig* le triplet (couleur de u, couleur de e, $rang[u]$).

Remarques :
- La couleur $\lambda(u)$ d'un sommet u correspond à la nature du motif chimique associé, et inclus ainsi le semi-degré sortant de u (i.e. le nombre de fonctions caractéristiques sur le motif chimique associé). On utilise ici la coloration des sommets pour obtenir une caractérisation des sommets plus forte que le degré (comme c'était le cas dans l'approche par partitionnement).

15. avec l'information topologique donnée sous la forme d'un ordre des arcs sortants autour des sommets, à permutation circulaire près.

— De même, la couleur $\lambda(e)$ d'un arc e correspond à la nature du groupe caractéristique associée, et permet de déduire la couleur de l'arc complémentaire e^c.
— On remarquera que ce parcours en profondeur, ne modifie la signature faible que pour les arcs «retours», diminuant ainsi la taille de la signature faible finale.
— La pile P est une file LIFO [16], le premier élément ajouté étant le dernier enlevé.

Lemme : *À la fin de l'algorithme, la seule donnée de sig permet de reconstruire la carte $\mathcal{M}$ donnée en entrée. Autrement dit, sig est une signature faible.*

Preuve : À chaque triplet $(\lambda(u), \lambda(e), rang[u])$ de la signature faible, on peut reconstruire la paire d'arcs (correspondant à une liaison chimique) empruntée lors du parcours en profondeur ($\lambda(e^c)$ étant entièrement définie par $\lambda(e)$).

De plus, on récupère le sommet u visité, avec une différentiation de chaque sommet grâce à l'utilisation du rang de u (i.e. le nombre d'étapes du parcours avant de découvrir le sommet u).

L'information obtenue sur la couleur du sommet u permet d'extraire les triplets correspondant aux voisins de u, puisque $\lambda(u)$ inclus le degré de u. Donc par récursivité, on récupère les voisins de chaque sommet, tous labellisés avec un identifiant unique provenant du rang.

Finalement, les voisins de chaque sommet u visité étant visités dans l'ordre donné autour de ce sommet, on peut reconstruire cet ordre pour chaque sommet visité. Notons, que cet ordre est retrouvé à permutation circulaire près, puisque l'on commence le parcours des arcs autour de u par un arc arbitraire.

Il reste alors à montrer que le parcours proposé couvre l'ensemble du graphe pour conclure.

C'est ici qu'intervient l'initialisation de la pile P. En effet, sur l'exemple figure 1, on constate qu'en partant du sommet u et de l'arc e, le parcours n'explorerait jamais la partie gauche du graphe si l'on n'ajoutait pas (u, e) dans P au tout début du parcours. Cependant, avec cet ajout, on est assuré de repasser par u, et ainsi d'explorer tout le graphe, comme pour un parcours en profondeur classique. En effet, on ne considère ici que des cartes connexes (voir section 1.2.4).

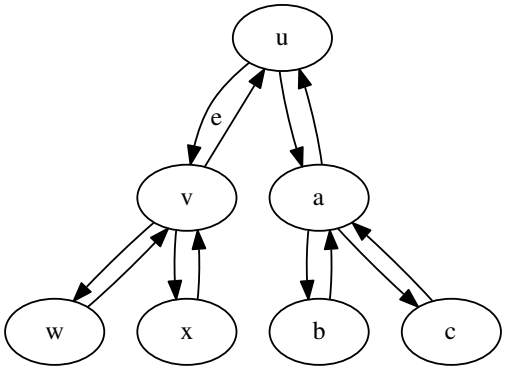

Figure 1 – Représentation d'un plongement sphérique où l'arc e est mis en valeur, tel qu'en partant du sommet u et de l'arc e, le parcours n'explorerait jamais la partie gauche du graphe si l'on n'ajoutait pas (u, e) dans P au tout début du parcours.

Ainsi, la signature obtenue par notre parcours est faible, car à un même graphe, on peut associer plusieurs parcours, en fonction du couple (u, e) initial choisi. Il est donc nécessaire d'assurer l'unicité de la signature $\sigma_0(\mathcal{M})$ d'une carte $\mathcal{M}$ donnée.

16. Last In, First Out.

Afin de choisir canoniquement la signature forte $\sigma_0(\mathcal{M})$ d'une carte $\mathcal{M}$ donné, on va sélectionner la signature faible minimale, pour l'ordre lexicographique, parmi l'ensemble des signatures faibles obtenables par notre parcours de $\mathcal{M}$:

$$\forall \mathcal{M} \quad \sigma_0(\mathcal{M}) = \min_{\text{lex}}(\{\text{DFS\_modif}(\mathcal{M}, u, e = (u, v)) \ ; \ u, v \in \text{V}\})$$

avec $\min_{\text{lex}}$ défini comme l'ordre lexicographique tel que, si on considère sig comme une suite de triplets $(t_n)_{n\in\mathbb{N}} = ((\lambda(u_n), \lambda(e_n), rang[u_n]))_{n\in\mathbb{N}}$, alors :

$$(t_n)_{n>i} <_{\text{lex}} (t'_{n'})_{n'>i} \quad \Leftrightarrow \quad t_i < t'_i \quad \vee \quad \big(t_i = t'_i \quad \wedge \quad (t_n)_{n>i+1} <_{\text{lex}} (t'_{n'})_{n'>i+1}\big)$$

Théorème : *σ_0 est une fonction de signature.*

Preuve : L'utilisation de l'ordre lexicographique $\min_{\text{lex}}$ permet d'assurer l'unicité et la canonicité de σ_0, grâce aux propriétés de cet ordre total.

Or, d'après le lemme, pour toute carte $\mathcal{M}$, $\sigma_0(\mathcal{M})$ est une signature faible.

Ainsi, par définition d'une signature forte, σ_0 est une fonction de signature, car pour toute carte $\mathcal{M}$, σ_0 fournit une unique signature faible, donc $\sigma_0(\mathcal{M})$ est une signature forte.

2.4.3 Analyse de complexité

Soit $\mathcal{M} = (\text{V}, \text{E})$ une carte.

DFS_modif$(\mathcal{M}, u, e = (u, v))$, avec $u, v \in \text{V}$, s'exécute en $O(|V| + |E|) = O(|E|)$ [17].

Or, pour calculer $\sigma_0(\mathcal{M})$, il faut tester tous arcs de départ possibles. Donc $\sigma_0(\mathcal{M})$ se calcule en $O(|E|^2)$.

En pratique, lors du calcul de $\sigma_0(\mathcal{M})$, on conserve la signature faible la plus petite (pour $\min_{\text{lex}}$) trouvée $\sigma_{\min}(\mathcal{M})$.

Ainsi, à chaque parcours, on s'arrête dès que l'on trouve que la signature faible $\sigma(\mathcal{M})$ en cours de calcul est plus grande que $\sigma_{\min}(\mathcal{M})$. En effet, dès que $\sigma(\mathcal{M})$ devient plus grand que $\sigma_{\min}(\mathcal{M})$ pour $\min_{\text{lex}}$, on peut conclure, sans avoir à calculer la fin de $\sigma(\mathcal{M})$ (car on utilise un ordre lexicographique, et $\sigma(\mathcal{M})$ est une concaténation de triplets).

On se référera par la suite à cette technique par l'appellation : couper la signature.

Par conséquent, cette technique permet d'améliorer en pratique la performance de l'algorithme utilisé, puisque tous les parcours ne sont pas forcément exécutés complètement.

Finalement, l'ensemble Σ des signatures fortes [18] calculées est stocké dans un AVL (arbre binaire de recherche équilibré). D'où une complexité globale en $O(|\mathcal{S}| \times (|\text{E}^2_{\max}| + |\text{E}_{\max}| \log |\mathcal{S}|))$.

En effet, la signature $\sigma_0(\mathcal{M})$ comporte au maximum $|\text{E}| - |\text{V}| + 2$ triplets. En effet, on ne stocke pas les arcs menant à un sommet non marqué (non déjà vu), mais on ajoute d'office les deux premiers éléments. Soit une comparaison de deux signatures en $O(|\text{E}_{\max}|)$.

Finalement, chaque signature correspondant à une classe d'isomorphismes, l'algorithme possède une complexité en espace en $O(|\mathcal{S}| \times (|\text{E}_{\max}|)$.

3 Optimisations

Le but du stage étant d'optimiser cet algorithme, différentes approches ont été étudiées. Ainsi, le but était de diminuer le nombre d'étapes nécessaires en moyenne pour le calcul d'une signature forte.

17. Car chaque motif chimique possède au moins un groupe caractéristique
18. Chaque signature forte correspond ici à une classe d'isomorphismes.

Pour y arriver, nous distinguerons deux méthodes majeures :
— Diminuer le nombre d'étapes par parcours, en enrichissant la signature, de manière à couper le calcul des signatures faibles plus tôt (voir section 3.1).
— Diminuer le nombre de parcours à tester, en sélectionnant les arcs de départ de manière plus contraignante (voir section 3.2).

3.1 Enrichissement de la signature

Méthode générale : Dans cette approche, on va chercher à couper plus tôt le calcul des signatures (voir section 2.4.3), c'est-à-dire différentier plus rapidement deux signatures différentes.

Chaque signature faible est constituée d'une suite d'éléments, comparés aux éléments d'une signature de référence $\sigma_{\min}$. Le calcul s'arrêtant dès que le dernier élément calculé est supérieur à l'élément de même indice dans la signature de référence.

Autrement dit, pour une carte $\mathcal{M}$ dont on a pour l'instant la signature la plus petite $\sigma_{\min}(\mathcal{M})$, le calcul d'une signature faible $\sigma(\mathcal{M})$ prend un nombre d'étapes nb_s définit par :

$$nb_s = \min(\{i,\ \sigma(\mathcal{M})[i] > \sigma_{\min}(\mathcal{M})[i]\} \cup \{|\sigma(\mathcal{M})|\})$$

On cherche alors ici à diminuer la taille du préfixe commun à deux signatures faibles d'une même carte $\mathcal{M}$. On va ainsi chercher à enrichir les éléments des signatures, afin de favoriser leur distinction.

Pour enrichir les éléments des signatures, on associe à chaque arc e une face f_e (définie de manière canonique), et on ajoute au triplet initial $(\lambda(u), \lambda(e), rang[u])$ la taille de f_e, pour donner $(\lambda(u), \lambda(e), rang[u], |f_e|)$.

Définition : La face f_e choisie pour un arc e est le cycle $(e_n)_n$ passant par e, tel que à l'étape i, $(u_{i+1}, e_{i+1}) = \texttt{suiv}(\texttt{comp}(u_i, e_i))$ avec la définition de `comp` et `suiv` de la section 2.4.2.

L'ensemble E des arcs de $\mathcal{M}$ est alors partitionnable en faces[19], et on est ainsi assuré de pouvoir repasser par e, grâce à la saturation de $\mathcal{M}$.

Justification : Le choix de la taille des faces pour l'enrichissement vient de la facilité de calcul de cette information et de son utilisation dans la littérature (c'est en effet, l'un des paramètres de la fonction de discrimination dans l'article exposé à la section 2.3.2).

La rapidité de calcul de cette information est crucial ici, puisqu'elle doit être calculée pour chaque carte. De plus, son temps de calcul doit être inférieur au gain apporté par cette méthode pour être réellement efficace.

Or, en utilisant le fait que E peut être partitionné en faces, on arrive à calculer f_e pour tout $e \in$ E en $O(|\text{E}|)$. Il suffit pour cela de prendre un arc e quelconque de $\mathcal{M}$, de parcourir f_e, puis mettre à jour les arcs visités à la fin du parcours de f_e, et de recommencer pour un arc non marqué, jusqu'à ce que tous les arcs soient marqués.

Ainsi, chaque arc n'a été parcouru qu'une fois, d'où une complexité en $O(|\text{E}|)$.

Quant à la pertinence de cette information, elle s'explique intuitivement par le fait que les cartes correspondantes aux cages moléculaires possèdent de nombreux cycles. Ainsi, l'ajout de la taille des faces permet d'apporter localement une information relativement globale.

Par conséquent, lors du calcul des signatures faibles, on espère acquérir rapidement (i.e. en ayant encore peu parcouru le graphe) une information supplémentaire plus globale permettant de discriminer des situations apparaissant localement semblables.

19. Une arête ne peut pas appartenir à deux faces différentes, car le parcours est déterministe et sans mémoire, donc canoniquement déterminé par la simple donnée d'une de ses arêtes.

Résultats : En pratique, le gain de cette méthode ne parvient souvent pas à compenser son coût supplémentaire.

En étudiant plus en détails la situation, il apparaît que cette méthode permet en effet de couper significativement plus tôt, lorsque il est possible de couper plus tôt.

Cependant, les parcours ne pouvant pas être coupés représentent la majeure partie du nombre d'étapes de parcours par calcul de signature forte, car le mécanisme de coupe est déjà très performant.

Ainsi, les améliorations d'un mécanisme déjà performant se retrouvent noyées dans les étapes non améliorables du calcul d'une signature forte.

En conclusion, le gain apporté par cette méthode n'est pas suffisant, et il y a peu de marge d'amélioration restante pour cette approche.

Par conséquent, une nouvelle approche a été étudiée.

3.2 Heuristiques

Méthode générale : Au lieu de chercher à couper plus tôt le calcul de signature faible, on va ici chercher à diminuer le nombre de signatures faibles à calculer pour déterminer la signature forte.

La signature forte $\sigma_0(\mathcal{M})$ d'une carte $\mathcal{M}$ est choisie canoniquement parmi l'ensemble $\mathrm{S}_\mathcal{M}$ des signatures faibles correspondants aux couples (sommet u, arc e) de départ possibles de $\mathcal{M}$. On notera que le couple (u, e) doit respecter la condition d'adjacence, à savoir, e doit être un arc sortant de u. Cette condition sera implicitement supposée dans toute la suite.

Un raffinement basique de l'ensemble des couples de départ est la sélection des couples (u, e) de départ tels que $\lambda(u)$ et $\lambda(e)$ soient arbitrairement fixés (en vérifiant leur compatibilité) dès le début du programme.

Cette méthode est utilisée par l'algorithme initial de ce projet, en sélectionnant le premier motif décrit en entrée, et le premier type d'arc décrit pour ce motif (voir l'annexe 7.2).

Cependant, il est facile de se convaincre que ce choix peut être grandement amélioré à faible coût. En effet, le nombre nb_c de couples (u, e), de couleurs $(\lambda(u), \lambda(e))$, présents dans une carte $\mathcal{M}$ vérifie :

$$nb_c = |\{\!\{v,\ \lambda(v) = \lambda(u) \wedge v \in \mathrm{V}\}\!\}| \times \gamma(\lambda(e), \lambda(u))$$

où $|\{\!\{v,\ \lambda(v) = \lambda(u) \wedge v \in \mathrm{V}\}\!\}|$ représente le nombre de sommets de $\mathcal{M}$ qui possède la même couleur que u (correspondant au type de motif associé), et $\gamma(\lambda(e), \lambda(u))$ représente la multiplicité des arcs de couleur $\lambda(e)$ sortant d'un sommet de couleur $\lambda(u)$.

Si le terme $|\{\!\{v,\ \lambda(v) = \lambda(u) \wedge v \in \mathrm{V}\}\!\}|$ dépend de la carte $\mathcal{M}$ étudiée, le terme $\gamma(\lambda(e), \lambda(u))$ ne dépend que de la définition des motifs chimiques donnés en entrée.

En effet, chaque couleur de sommet correspond à un type de motif de base, défini entre autres par les types et le nombre de ses arcs sortants (correspondant aux groupes caractéristiques chimiques associées, fixées pour un même type de motif chimique). Ainsi, on a accès à la valeur $\gamma(\lambda(e), \lambda(u))$ pour tout $\lambda(u)$ et $\lambda(e)$, restant valable pour toute carte $\mathcal{M} \in \mathcal{S}$ considérée.

3.2.1 Optimisation statique

Cette indépendance de $\gamma(\lambda(e), \lambda(u))$ par rapport à $\mathcal{M}$ permet alors d'optimiser le choix des couleurs $\lambda_{\mathrm{opt}}(e)$ et $\lambda_{\mathrm{opt}}(u)$ tels que $\gamma(\lambda_{\mathrm{opt}}(e), \lambda_{\mathrm{opt}}(u))$ soit minimal, afin de raffiner plus intelligemment l'ensemble des couples (u,e) de départ possibles, et donc réduire globalement l'ensemble $\mathrm{S}_\mathcal{M}$ de signatures faibles nécessaires au calcul de $\sigma_0(\mathcal{M})$ pour une carte $\mathcal{M}$ quelconque.

De plus, ce simple calcul de $\lambda_{\text{opt}}(e)$ et $\lambda_{\text{opt}}(u)$ étant effectué une seule fois dans l'exécution du programme, le surcoût occasionné est clairement négligeable.

Cependant, il est préférable de s'assurer que le choix de $\lambda_{\text{opt}}(e)$ et $\lambda_{\text{opt}}(u)$ est canonique, afin de conserver les mêmes performances entre plusieurs exécutions du programme.

Pour cela, nous choisissons parmi les couples de couleurs possibles, ceux dont le sommet est de plus haut degré, intuitivement car il est moins probable d'avoir une faible multiplicité sur un sommet de haut degré. Le choix final reposant sur l'ordre dans lequel les motifs sont indiqués dans le fichier d'entrée.

L'optimisation statique possède néanmoins un inconvénient majeur. La couleur du sommet $\lambda_{\text{opt}}(u)$ choisie peut ne pas être présente dans une carte $\mathcal{M}$, en toute généralité, si elle n'est pas nécessaire. Par conséquent, pour utiliser cette méthode il est nécessaire de s'assurer de la nécessité de chaque type de sommet.

Résultats : En pratique, cette méthode apporte un gain non négligeable (pouvant facilement diminuer de moitié le temps de calcul global, en fonction de l'ordre des motifs décrits en entrée), pour un coût quasi nul.

De plus, cette méthode permet de s'abstraire partiellement de l'ordre dans lequel les motifs sont décrits en entrée. Cela est particulièrement appréciable, en particulier pour un programme s'adressant en priorité à les chimistes non spécialistes dans le calcul d'isomorphisme.

3.2.2 Optimisation dynamique

Vu les gains apportés par l'optimisation classique, nous avons étudié une autre méthode plus poussée, visant à diminuer le nombre de signatures faibles à calculer pour déterminer la signature forte d'une carte $\mathcal{M}$.

Contrairement à l'heuristique statique qui fixe $\lambda_{\text{opt}}(e)$ et $\lambda_{\text{opt}}(u)$ indépendamment des cartes $\mathcal{M}$ considérées, en optimisant uniquement le terme $\gamma(\lambda_{\text{opt}}(e), \lambda_{\text{opt}}(u))$, on optimise ici $\lambda_{\text{opt}}(e)$ et $\lambda_{\text{opt}}(u)$ pour chaque carte afin de diminuer nb_c.

Ainsi, on est assuré de réduire au maximum l'ensemble $\mathrm{S}_{\mathcal{M}}$ de signatures faibles nécessaires au calcul de $\sigma_0(\mathcal{M})$ pour chaque carte $\mathcal{M}$, en discriminant sur les couleurs des sommets et arcs.

Pour cela, il est nécessaire de calculer pour chaque carte $\mathcal{M}$ la multiplicité de chaque type (ou couleur) de sommet. En effet, le terme $\gamma(\lambda_{\text{opt}}(e), \lambda_{\text{opt}}(u))$ est optimisé au début de l'algorithme, et indépendant de $\mathcal{M}$. Quant au terme $|\{\!\{v,\ \lambda(v) = \lambda(u) \wedge v \in \mathrm{V}\}\!\}|$, il ne dépend pas de la répartition des arcs.

On peut ainsi optimiser les paramètres $\lambda_{\text{opt}}(e)$ et $\lambda_{\text{opt}}(u)$ en s'intéressant à un seul degré de liberté, $\lambda_{\text{opt}}(u)$. [20]

Pour cela, il suffit de conserver un tableau associant pour chaque couleur de sommet, la couleur d'arc la moins représentée autour de ce sommet, ainsi que sa multiplicité.

Ainsi, il suffit de compter les occurrences de chaque type de sommets présents dans $\mathcal{M}$, et de les multiplier par la multiplicité minimale d'arc associée, puis de sélectionner le minimum. L'unicité finale étant assurée par l'ordre de définition des motifs en entrée.

20. L'approche glouton présentée est ainsi optimale.

Résultats : En pratique, le surcoût supplémentaire est presque négligeable, car inclus dans la boucle du pré-calcul des triplets (récupération des couleurs).

Ainsi, cette heuristique permet d'obtenir un gain non négligeable dans certains cas (on peut construire des exemples de motifs où le gain apporté est aussi grand que souhaité), pour un surcoût négligeable.

De plus, cette méthode permet également de s'abstraire partiellement de l'ordre dans lequel les motifs sont décrits en entrée, comme précédemment.

3.3 Cumul

Méthode générale : On va ici cumuler les deux méthodes en réintroduisant le calcul des faces détaillé dans la section 3.1.

La performance des heuristiques permet en effet d'envisager la réintroduction du calcul des faces, malgré son surcoût non négligeable.

En effet, l'heuristique dynamique (détaillée à la section 3.2.2) permet déjà un raffinement maximal sur les couleurs des sommets et des arcs. Ainsi, pour voir une meilleure heuristique, il est nécessaire d'introduire une donnée supplémentaire.

On étudie ici l'ajout de la taille des faces (voir section 3.1), à la fois comme enrichissement de la signature et comme heuristique. L'utilisation comme heuristique étant implémentés selon deux méthodes de raffinement distinctes.

3.3.1 Approche glouton

Afin de faciliter la recherche du triplet $(\lambda(u), \lambda(e), |f_e|)$ le moins représenté pour chaque carte $\mathcal{M}$, on optimise séparément $(\lambda(u), \lambda(e))$ selon la méthode décrite à la section 3.2.2, puis on optimise $|f_e|$ parmi des couples (u, e) de couleurs $(\lambda(u), \lambda(e))$.

Par conséquent, on obtient une approximation du triplet $(\lambda(u), \lambda(e), |f_e|)$ le moins représenté, en $O(|\mathrm{V}| + |\mathrm{E}|)$, car $0 < |f_e| \leq |\mathrm{E}|$.

Résultats : En pratique, le surcoût du calcul des faces reste important, mais le gain associé à son utilisation en tant qu'heuristique permet de rendre cette méthode comparable aux approches précédentes sauf, bien évidemment, dans les graphes suffisamment réguliers pour avoir des cycles de même taille (ce qui peut être en réalité assez commun en chimie, pour des raisons énergétiques, selon les modélisations des motifs).

3.3.2 Approche optimale

On cherche ici directement le triplet $(\lambda(u), \lambda(e), |f_e|)$ le moins représenté pour chaque carte $\mathcal{M}$, sans approximation.

La recherche des triplets les moins représentés se fait en ne parcourant que les triplets présents dans C. En effet, en pratique peu de combinaisons de triplets sont représentés, ce qui revient à chercher un minimum dans un tableau creux [21].

Résultat : En pratique, le coût supplémentaire de la recherche est en partie noyé dans le surcoût lié au calcul des tailles de faces.

Ainsi, on est assuré d'avoir une heuristique optimale sur les tailles de faces, les couleurs de sommets et d'arcs, pour un surcoût relativement faible.

21. Un tableau dont les données non nulles (ici intéressantes) sont en faible proportion.

4 Pré-calcul

On étudie ici une approche alternative pour partitionner l'ensemble $\mathcal{S}$ des cartes en classes d'isomorphismes.

4.1 Principe

Dans l'approche initiale, la signature forte $\sigma_0(\mathcal{M})$ d'une carte $\mathcal{M}$ est définie par :

$$\forall \mathcal{M} \quad \sigma_0(\mathcal{M}) = \min_{\text{lex}}(\{\text{DFS\_modif}(\mathcal{M}, u, e = (u, v)) \ ; \ u, v \in \text{V}\}) = \min_{\text{lex}}(\text{S}_{\mathcal{M}})$$

avec $\text{S}_{\mathcal{M}}$ l'ensemble des signatures faibles de $\mathcal{M}$.

Cependant, on peut également considérer l'ensemble $\text{S}_{\mathcal{M}}$ comme étant une signature forte de $\mathcal{M}$, la condition d'unicité étant immédiatement vérifiée.

Théorème : Pour comparer si deux cartes $\mathcal{M}$ et $\mathcal{M}'$ sont isomorphes, il suffit de tester l'appartenance d'une signature faible $\sigma'(\mathcal{M}')$ de $\mathcal{M}'$ dans l'ensemble $\text{S}_{\mathcal{M}}$.

Preuve : Par définition d'un signature faible, si l'on trouve une signature faible $\sigma(\mathcal{M}) \in \text{S}_{\mathcal{M}}$ de $\mathcal{M}$ telle que $\sigma(\mathcal{M}) = \sigma'(\mathcal{M}')$, alors $\mathcal{M}$ est isomorphe à $\mathcal{M}'$.

Réciproquement, si $\sigma'(\mathcal{M}') \notin \text{S}_{\mathcal{M}}$, alors par définition de $\text{S}_{\mathcal{M}'}$, $\sigma'(\mathcal{M}') \in \text{S}_{\mathcal{M}'}$, et par conséquent $\text{S}_{\mathcal{M}} \neq \text{S}_{\mathcal{M}'}$. Donc par définition d'une signature forte, $\mathcal{M}$ et $\mathcal{M}'$ ne sont pas isomorphes.

On exploite alors ce théorème pour partitionner efficacement l'ensemble $\mathcal{S}$, en particulier lorsque la proportion de copies isomorphes est importante.

Pour chaque carte $\mathcal{M}$, on calcule une signature faible $\sigma(\mathcal{M})$ possible.

On teste alors l'appartenance de $\sigma(\mathcal{M})$ pour chaque ensemble $\text{S}_{\mathcal{M}'}$ déjà trouvés (chaque ensemble $\text{S}_{\mathcal{M}'}$ correspondant à une classe d'isomorphismes).

S'il existe $\mathcal{M}'$ tel que $\sigma(\mathcal{M}) \in \text{S}_{\mathcal{M}'}$, alors $\mathcal{M}$ est une copie isomorphe à $\mathcal{M}'$, et on passe à une carte suivante de $\mathcal{S}$.

Sinon, $\mathcal{M}$ appartient à une nouvelle classe d'isomorphisme, et la signature forte (l'ensemble $\text{S}_{\mathcal{M}}$) de $\mathcal{M}$ est calculée et stockée.

Ainsi, cette approche en $O\left(|\text{E}||\mathcal{S}| + |\text{E}| \times \sum_{\text{S}_{\mathcal{M}} \in \{\text{S}_{\mathcal{M}'},\ \mathcal{M}' \in \mathcal{S}\}} |\text{S}_{\mathcal{M}}|\right)$ (sans compter le temps de comparaison) est d'autant plus efficace que la proportion de copies isomorphes est grande. Puisque chaque copie isomorphe ne nécessite le calcul que d'une signature faible (terme en $O(|\text{E}||\mathcal{S}|)$), alors que chaque classe d'isomorphisme nécessite le calcul de toutes les signatures faibles possibles (terme en $O\left(|\text{E}| \times \sum_{\text{S}_{\mathcal{M}} \in \{\text{S}_{\mathcal{M}'},\ \mathcal{M}' \in \mathcal{S}\}} |\text{S}_{\mathcal{M}}|\right)$), sans possibilité de couper la signature [22].

À cause du compromis espace-temps utilisé (signature forte plus volumineuse), la quantité de signatures à comparer est plus important. Ainsi, l'efficacité de cette technique est plus fortement influencée par la rapidité de comparaison des signatures faibles, d'où l'intérêt d'avoir une signature petite, et peu de signatures faibles possibles.

Ce résultat suggère l'application d'une heuristique à cette méthode pour accroître sa performance. Par contre, l'enrichissement de la signature n'aurait aucune utilité ici, puisque l'on ne coupe pas le calcul des signatures faibles.

22. Par conséquent, il ne sert à rien d'enrichir la signature

4.2 Heuristiques

Ainsi, les heuristiques présentées dans la section 3.2 ont été appliqués à cette méthode.

L'effet des heuristiques est moins flagrant qu'avec l'approche initiale, car pour chaque copie isomorphe, une seule signature faible est calculée, peu importe l'heuristique appliquée.

Par conséquent, dans le cas de l'heuristique dynamique, on a un surcoût pour chaque carte $\mathcal{M}$ certes négligeable, mais pour un gain nul lorsque $\mathcal{M}$ appartient à l'une des classes d'isomorphismes déjà vues.

Résultats : Couplée avec une heuristique, cette approche est la plus efficace actuellement. En effet, elle tire profit de l'importante proportion de copies isomorphes.

En pratique, le facteur limitant est désormais dans de nombreux cas la comparaison et le stockage des signatures dans l'AVL. Cela suggère la nécessité de changer de structure de données pour accroître les performances.

5 Résultats expérimentaux et conclusions

Examinons en détails quelques résultats expérimentaux, avant de récapituler les résultats importants, pour enfin détailler les pistes d'améliorations futures.

5.1 Comparaisons expérimentales

Les exemples étudiés dans cette section correspondent à la génération de cartes planaires issus de motifs prédéfinis tels que le nombre de chaque type de sommet ne varie pas d'une carte à l'autre.

5.1.1 Heuristique statique optimale

L'exemple illustré sur la figure 2 représente le cas où l'heuristique statique et l'heuristique dynamique donnent le même résultat pour chaque carte $\mathcal{M}$.

On peut constater de manière générale que :

- L'enrichissement de la signature seul (version 2.0) n'est pas assez efficace pour compenser le surcoût associé, malgré la faible réduction du nombre d'étapes par parcours.
- Si les heuristiques diminuent évidemment significativement le nombre de signatures faibles (i.e. le nombre de parcours) par calcul de signature forte (en diminuant le nombre de points de départ possibles), cela se traduit par une augmentation du nombre d'étapes moyens par parcours. Cela est lié au fait que les parcours pouvant être coupés sont moins nombreux, alors qu'il y aura toujours au moins un parcours ne pouvant pas être coupé.
- La différence entre le nombre d'étapes de parcours par calcul de signature forte et le temps d'exécution total est en général due aux coûts des pré-calculs (ou temps de comparaison pour la version 5).

Le gain apporté par le pré-calcul dans le cas des heuristiques dynamiques par rapport aux heuristiques statique est ici nul (on le voit clairement dans l'égalité du nombre de parcours possibles par calcul de signature forte). Ainsi, il est plus facile d'apprécier le caractère presque négligeable de ce surcoût.

5.1.2 Heuristique statique pire cas

L'exemple illustré sur la figure 3 représente le cas contraire où l'heuristique statique est toujours moins bonne que l'heuristique dynamique, pour chaque carte $\mathcal{M}$. Ici, l'heuristique statique fournit 2 fois plus de points de départs possibles qu'avec l'heuristique dynamique.

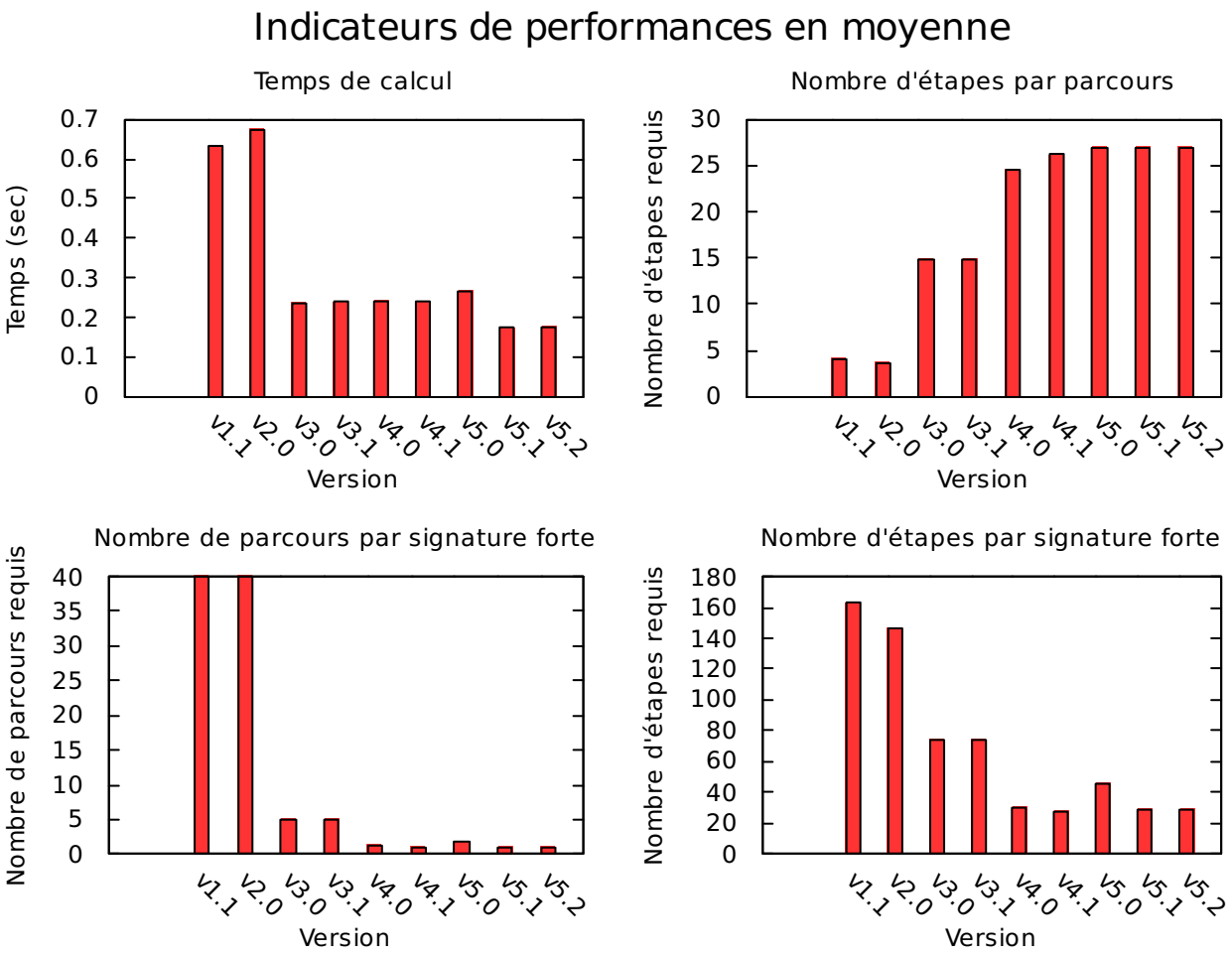

Figure 2 – Représentation graphique de quelques indicateurs de performance (temps de calcul global, nombre moyen d'étapes du parcours utilisé pour calculer les signatures faibles, nombre moyen de parcours nécessaires pour calculer une signature forte, nombre moyen d'étapes de parcours pour calculuer une signature forte) en fonction de la version de l'algorithme étudié. Version 1.1 : algo initial (sans heuristique basique) ; version 2.0 : enrichissement de la signature ; version 3.0 : heuristique statique ; 3.1 : heuristique dynamique ; 4.0 : cumul glouton ; 4.1 : cumul optimal ; 5.0 : précalcul ; 5.1 : précalcul + heuristique statique ; 5.2 : précalcul + heuristique dynamique.

En plus des constatations générales décrites dans l'exemple précédent, et justifiées dans les sections 3 et 4, on constate ici que la réduction du nombre de points de départ possibles impacte significativement les performances, ainsi que le nombre d'étapes de parcours par calcul de signature forte.

On remarque également que cet effet est plus flagrant lorsque appliqué à la méthode initiale (version 3.0 et 3.1) qu'à la méthode prônant le pré-calcul exhaustif des signatures faibles (version 5.0 et 5.1).

Cela se comprend aisément par le fait que les heuristiques apportent un gain nul lorsque la carte $\mathcal{M}$ étudiée est la copie isomorphe d'une carte $\mathcal{M}'$ déjà analysée.

5.2 Conclusion

Les études menés au cours de ce stage ont permis de montrer que des améliorations significatives de la méthode décrite dans la section 2.4 peuvent être apportées et implémentées simplement.

De plus, une comparaison des différentes améliorations a permis de dégager les situations dans lesquelles elles sont le plus adaptées, à travers l'identification des raisons d'échecs.

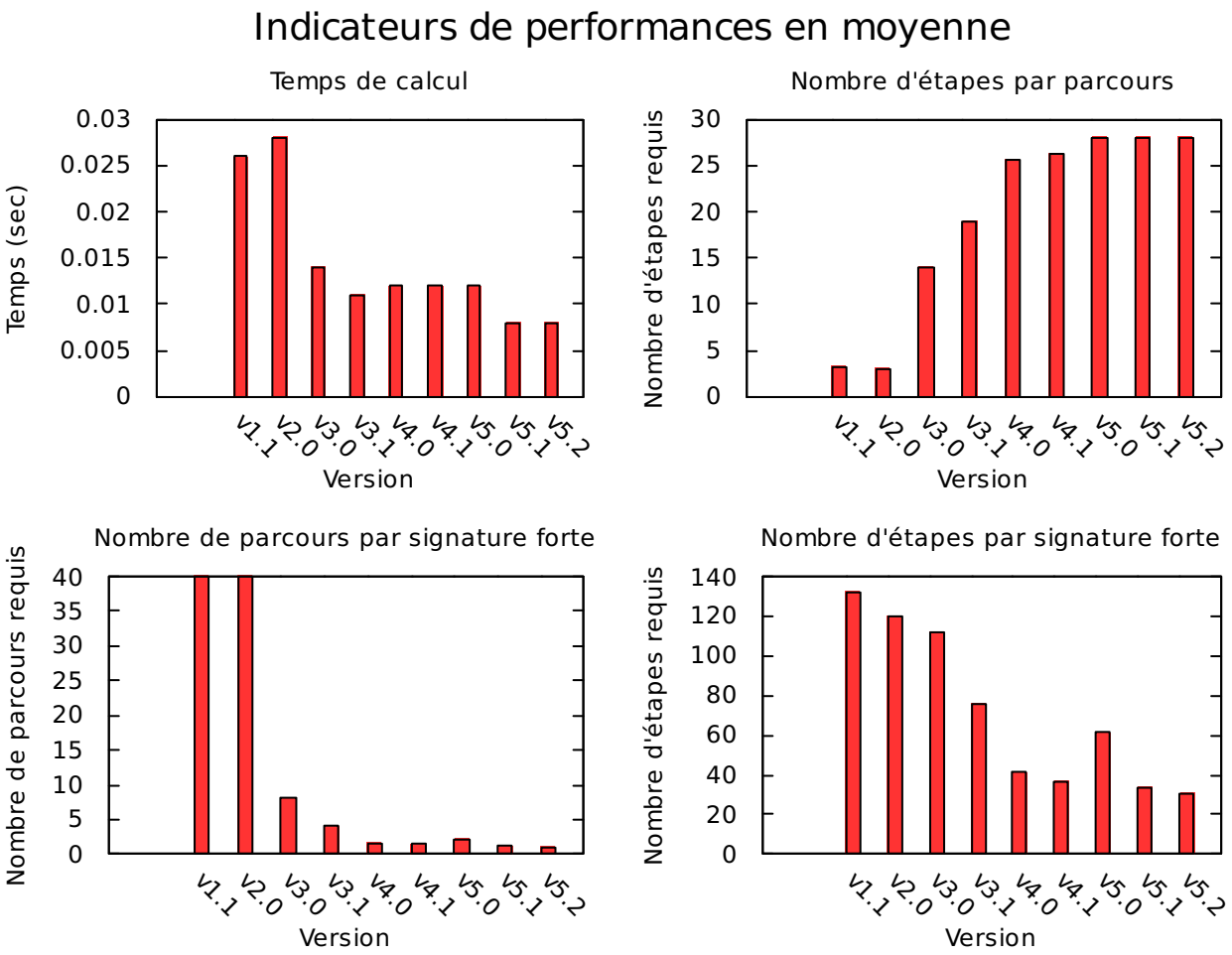

Figure 3 – Représentation graphique de quelques indicateurs de performance (temps de calcul global, nombre moyen d'étapes du parcours utilisé pour calculer les signatures faibles, nombre moyen de parcours nécessaires pour calculer une signature forte, nombre moyen d'étapes de parcours pour calculuer une signature forte) en fonction de la version de l'algorithme étudié. Version 1.1 : algo initial (sans heuristique basique) ; version 2.0 : enrichissement de la signature ; version 3.0 : heuristique statique ; 3.1 : heuristique dynamique ; 4.0 : cumul glouton ; 4.1 : cumul optimal ; 5.0 : précalcul ; 5.1 : précalcul + heuristique statique ; 5.2 : précalcul + heuristique dynamique.

Plus important encore, cette analyse critique a permis l'identification de certaines voies d'optimisations, et surtout d'exclure certaines, condamnées à échouer (comme par exemple l'enrichissement supplémentaire de la signature faible).

Finalement, une approche alternative tirant parti des spécificités des instances étudiées (forte proportion de copies isomorphes) a été développée et implémentée, dont la performance en moyenne n'est en pratique plus le facteur limitant.

5.3 Perspectives

Cependant, plusieurs voies d'amélioration sont en cours d'exploration :

— Un changement de structure de données plus adapté à la gestion d'un grand nombre de signatures faibles est actuellement en cours d'implémentation dans le laboratoire.

— La génération exhaustive des proportions de chaque type de sommet envisageables pour une taille de cage fixée, permettant de déterminer une heuristique statique plus intelligente est en cours d'implémentation.

— La comparaison aux librairies de références (Nauty et Traces)[12].
— L'élimination en amont des cages présentant une sous-partie peu stable chimiquement est un exemple de voie d'amélioration pouvant faire le but d'un projet plus conséquent, parmi de nombreuses autres optimisations nécessitant une connaissance approfondie et une étude poussée des caractéristiques chimiques des objets étudiés.

Remerciements

Je tiens à remercier tout particulièrement mon encadrant de stage Yann Strozecki pour son soutien et son encadrement d'une qualité exceptionnelle, mais également pour sa pédagogie, ses encouragements continuels, sa compréhension lors des mouvements de grève et des intempéries, et surtout sa passion contagieuse des énigmes informatiques et mathématiques.

Je tiens également à remercier chaleureusement Franck Quessette pour ses encouragements remarquables, sa patience, son sens de l'humour, et surtout pour m'avoir intégré spontanément au sein d'un projet tout aussi passionnant, mais que je n'ai malheureusement pas eu le temps de poursuivre durant mon stage.

Je tiens enfin à remercier Sandrine Vial pour son accueil convivial, et toute l'équipe MAGMAT pour l'ambiance chaleureuse du laboratoire.

Merci enfin à Carine Séraphim pour m'avoir supporté lors des phases de débogage.

6 Références

Références

[1] László Babai. Graph Isomorphism in Quasipolynomial Time. *arXiv :1512.03547 [cs, math]*, December 2015. arXiv : 1512.03547.

[2] H. S. Baird and Y. E. Cho. An Artwork Design Verification System. In *Proceedings of the 12th Design Automation Conference*, DAC '75, pages 414–420, Piscataway, NJ, USA, 1975. IEEE Press.

[3] Dominique Barth, Boubkeur Boudaoud, François Couty, Olivier David, Franck Quessette, and Sandrine Vial. Map Generation for CO 2 Cages. In Erol Gelenbe and Ricardo Lent, editors, *Computer and Information Sciences III*, pages 503–510. Springer London, London, 2013.

[4] Dominique Barth, Olivier David, Franck Quessette, Vincent Reinhard, Yann Strozecki, and Sandrine Vial. Efficient Generation of Stable Planar Cages for Chemistry. March 2015.

[5] Robert H. Crabtree, Maryellen. Lavin, and L. Bonneviot. Some molecular hydrogen complexes of iridium. *J. Am. Chem. Soc.*, 108(14) :4032–4037, July 1986.

[6] Roger L. DeKock and Wayne B. Bosma. The three-center, two-electron chemical bond. *J. Chem. Educ.*, 65(3) :194, March 1988.

[7] Stephen Heller, Alan McNaught, Stephen Stein, Dmitrii Tchekhovskoi, and Igor Pletnev. InChI - the worldwide chemical structure identifier standard. *Journal of Cheminformatics*, 5(1) :7, 2013.

[8] J. E. Hopcroft and J. K. Wong. Linear time algorithm for isomorphism of planar graphs (Preliminary Report). pages 172–184. ACM Press, 1974.

[9] J.E. Hopcroft and R.E. Tarjan. A V log V algorithm for isomorphism of triconnected planar graphs. *Journal of Computer and System Sciences*, 7(3) :323–331, June 1973.

[10] Hartmuth C. Kolb, M. G. Finn, and K. Barry Sharpless. Click Chemistry : Diverse Chemical Function from a Few Good Reactions. *Angewandte Chemie International Edition*, 40(11) :2004–2021, June 2001.

[11] Eugene M. Luks. Isomorphism of graphs of bounded valence can be tested in polynomial time. *Journal of Computer and System Sciences*, 25(1) :42–65, August 1982.

[12] Brendan D. McKay and Adolfo Piperno. Practical graph isomorphism, II. *Journal of Symbolic Computation*, 60 :94–112, January 2014.

[13] Regina de Cássia Nandi and André Luiz Pires Guedes. Graph Isomorphism applied to Fingerprint Matching. In *ResearchGate*, November 2006.

[14] Jeff Racine. gnuplot 4.0 : a portable interactive plotting utility. *Journal of Applied Econometrics*, 21(1) :133–141, January 2006.

[15] L. Weinberg. A Simple and Efficient Algorithm for Determining Isomorphism of Planar Triply Connected Graphs. *IEEE Transactions on Circuit Theory*, 13(2) :142–148, 1966.

[16] David Weininger. SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. *Journal of Chemical Information and Modeling*, 28(1) :31–36, February 1988.

7 Annexes

7.1 Contexte institutionnel

Le laboratoire DAVID, créé en juillet 2015 est un laboratoire d'Informatique de l'UFR des Sciences de l'Université de Versailles Saint-Quentin-en-Yvelines, dont les thématiques de recherche sont les Bases de Données et l'Algorithmique appliqués aux problématiques de ville intelligente et durable.

Le laboratoire DAVID regroupe 4 équipes de recherche, dont l'équipe MAGMAT que j'ai rejoint au cours de ce stage, sous la supervision de Yann Strozecki, maître de conférence à l'Université de Versailles Saint-Quentin.

L'équipe MAGMAT travaille en collaboration sur différents projets nationaux et européens, et en particulier une thématique «Modélisation et analyse moléculaire pour la santé» étudiée à l'aide de l'algorithmique discrète, l'un des domaines d'expertise de cette équipe.

Ainsi, mon stage s'inscrit ainsi principalement dans la sous-thématique «Conception de cages catalytiques bio-inspirées», issue de la chimie organique pour la chimie verte.

Le projet sur lequel j'ai travaillé durant ce stage est ainsi une collaboration avec le Groupe Synthèse et Réactivité de l'Institut Lavoisier Versailles.

Au cours de ce stage, j'ai ainsi fortement interagi avec les membres de l'équipe MAGMAT dirigé par Dominique Barth, et en particulier avec Yann Strozecki, Franck Quessette et Sandrine Vial.

7.2 Précisions sur le format d'entrée

Le programme développé au Laboratoire DAVID de l'UVSQ prend en entrée :
- La taille des cages moléculaires souhaitées, c'est-à-dire le nombre de motifs chimiques de base par cage.
- Un fichier de configuration d'extension .mot indiquant les motifs chimiques de base souhaités.

Ce fichier de configuration est défini de la manière suivante :
- La première ligne contient le nombre de types de motifs à définir, et un paramètre devenu caduque au fil des versions de développement.
- Les lignes qui suivent définissent les caractéristiques des motifs, chaque ligne représentant un motif, selon le schéma suivant : [Lettre identifiant le motif (sous la forme de sa position dans l'alphabet latin)] [Nombre de groupes caractéristiques (degré sortant du sommet associé)] [Liste

de la nature des groupes caractéristiques donnée dans le sens direct (liste de la couleur des arcs sortants)][23]

Exemple :

```
3 7
25 3 1 1 1
12 3 -1 1 2
10 3 1 -1 2
9 2 -2 -1
```

Ce fichier définit 4 types de motifs :

- Un motif représenté par la lettre Y ayant 3 groupes caractéristiques de même nature (ne pouvant s'associer qu'avec des groupes caractéristiques de label -1).
- Un motif L ayant 3 groupes caractéristiques différentes dont deux compatibles entre elles.
- Un motif J miroir de L. On remarque que l'ordre des couleurs a une importance ici, car L et J sont différents.
- Un motif I ayant deux groupes caractéristiques différentes et non compatibles.

23. La couleur d'un arc est représentée par un entier relatif, et sa couleur complémentaire est son opposé.