Projet molécule

Groupe A

2025 - 2026

Résumé

L'objectif de ce rapport est de présenter les différentes étapes du projet qui vise à étudier la similarité entre deux molécules.

Table des matières

1 Introduction

L'objectif principal de ce projet est le développement d'un programme capable de déterminer la similarité entre deux molécules en utilisant l'algorithme de McKay. En effet, la similarité entre molécules est une notion cruciale en chimes, et qui n'est pas triviale à définir. Nous souhaitons donc implémenter une solution efficace pour comparer des structures moléculaires et qui pourra, dans le futur servir dans divers domaines.

Dans un prmeier temps, nous allons nous concentrer sur la mise en place de l'algorithme de similarité, en utilisant des données moléculaires provenant de la base ChEBI.

Ensuite nous allons étudier les notions de distance entre molécules et implémenter un algorithme de clustering pour regrouper les molécules similaires.

Ce projet est à réaliser en groupe, ce qui porte une attention particulière à l'organisation et à la répartition des tâches entre les membres.

2 Organisation du groupe et planning des tâches

Le projet est un projet qui vise à mettre en avant le travail en équipe. Nous avons décidé de répartir les membres dans différentes tâches préalablement définies. Voici la répartition des tâches :

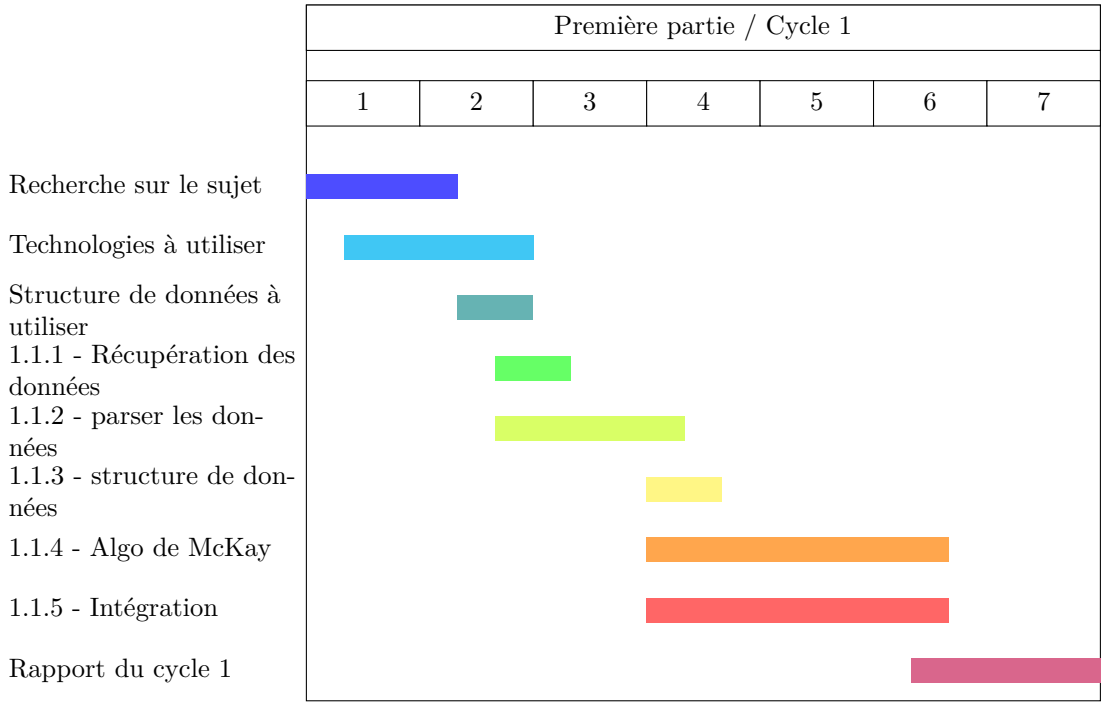

FIGURE 1 – Diagramme de Gantt — Cycle 1

Cette répartition permet à chaque membre de se concentrer sur une ou plusieurs tâches spécifiques, en s'assurant que chaque étapes est respecté. Ce la permet aussi de fixer des objectif à court et moyen terme, au lieu de seulement voir la deadline finale.

3 Étape 1 : récupération des données

2 options ont été considérées pour la récupération des données moléculaires :

— Récupération de plusieurs fichiers .mol via des requêtes HTTP vers l'API de ChEBI. L'objectif ici est de demander les motifs ChEBI des molécules via CMD, et de stocker les fichiers récupérés dans un dossier local. Les URL sont stockées dans un JSON pour faciliter la gestion des appels et l'entrée utilisateur.

Fichier JSON correspondant :

```
{
  "url_request_motif_part1": "https://www.ebi.ac.uk/chebi/backend/api/public/es_s
  "url_request_motif_part2": "&page=1&size=15",
  "url_request_id": "https://www.ebi.ac.uk/chebi/backend/api/public/molfile/"
}
```

On effectue une première requête avec les motifs et on récupère les ID des molécules demandées.

```
motif_id = data1["results"][0]["_id"]
```

Ensuite, pour cahque ID récupérés, on effecutue une seconde requete pour télécharger les .mol dans un dossier local.

— On télécharge un SDF complet depuis ChEBI, qui contient toutes les molécules que l'utilisateur donne. L'avantage est que l'on fait une seule requête HTTP, mais le fichier peut être plus lourd à gérer. Exemple d'une URL pour traiter 20 molécules :

```
url_sdf = "https://pubchem.ncbi.nlm.nih.gov/rest/pug/compound/cid
    /1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20/SDF"
```

Et on traite le fichier SDF pour extraire les molécules.

4 Étape 2 : Parser les données

Pour ce qui est de parser les données, nous avons aussi exporer deux options :

— Utilisation de RDKit, une bibliothèque Python spécialisée dans la chimie computationnelle. Ici on import Chem ce qui permet de lire les fichiers .mol et de manipuler les molécules très facilement. On peut ainsi parcourir pour chaque fichier .mol les atomes et les liaisons.

```
import sys
from rdkit import Chem

def export_to_graph(mol_path, out_path):
    mol = Chem.MolFromMolFile(mol_path)
    if not mol: return

    atoms = mol.GetAtoms()
    bonds = mol.GetBonds()

    num_atoms = len(atoms)
    num_bonds = len(bonds)
    total_nodes = num_atoms + num_bonds

    total_edges = num_bonds * 2
```

On traite ensuite les atomes et les liaisons pour créer le fichier de sortie. Le fichier de sortie est fichier .graph qui sera utilisé pour l'algorithme de McKay. En effet, chaque atome et chaque liaison devient un nœud dans le graphe et correspond parfaitement avec ce que demande Nauty.

— Écriture d'un parser personnalisé pour lire les fichiers .mol. Cette approche est plus complexe, mais elle permet une meilleure compréhension du format des fichiers .mol et nous permet de moduler le parser selon nos besoins spécifiques.

–> A DEVELOPPER SI ON CHOISIT CETTE OPTION

5 Étape 3 : Structure de données

6 Étape 4 : Algorithme de McKay

7 Étape 5 : Intégration