



Encodage des structures sous forme de graphe d'interactions et recherche de motifs



Réalisé par :

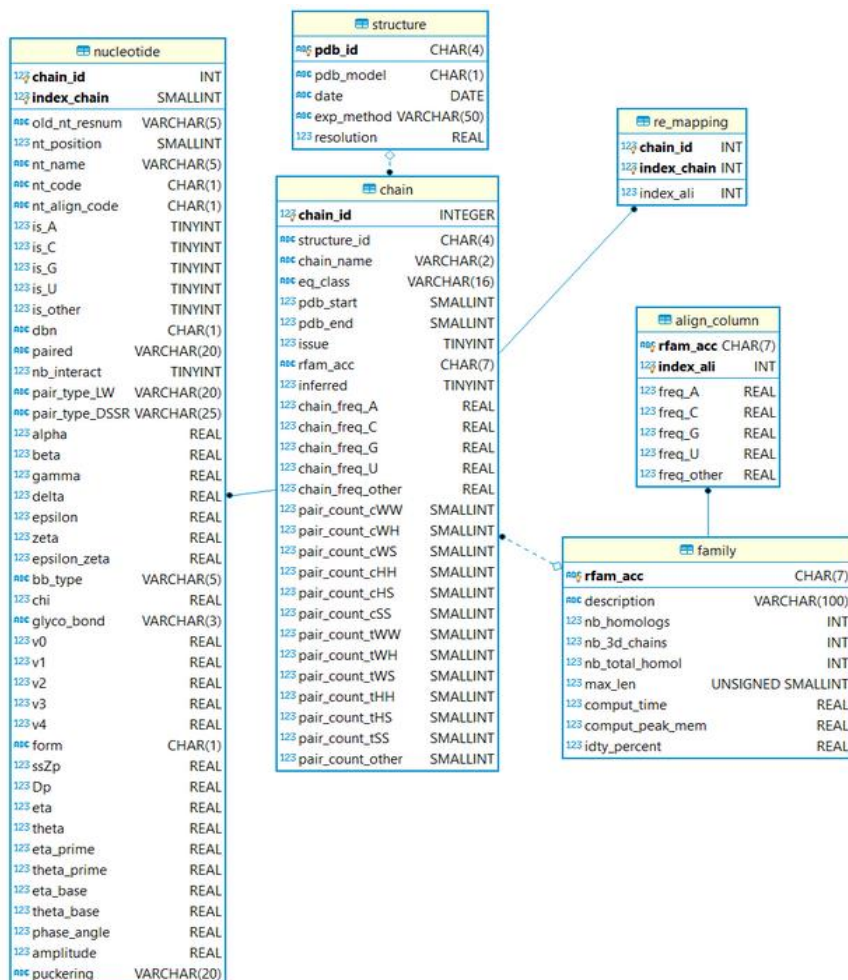
- AMMY DRISS SOUFIANE.
- ASSABBANE MEHDI.
- CHOUBBY IBTISSAM.

1. Contexte du projet

- Dans le cadre du Module de Bioinformatique, nous avons comme projet la réalisation d'une application qui permet :
 - L'extraction des données depuis une base de données SQLITE ou bien depuis des fichiers d'extension csv.
 - Exploiter ces données afin de les représenter sous forme de graph.
 - Recherche et détection des motifs dans une chaîne.

2. Description des données

- On dispose d'une base de données SQLITE, composé de plusieurs table, mais nous n'allons utiliser que les deux tables **chain** et **nucleotide** qui contiennent les données dont on aura besoin pour réaliser notre projet.

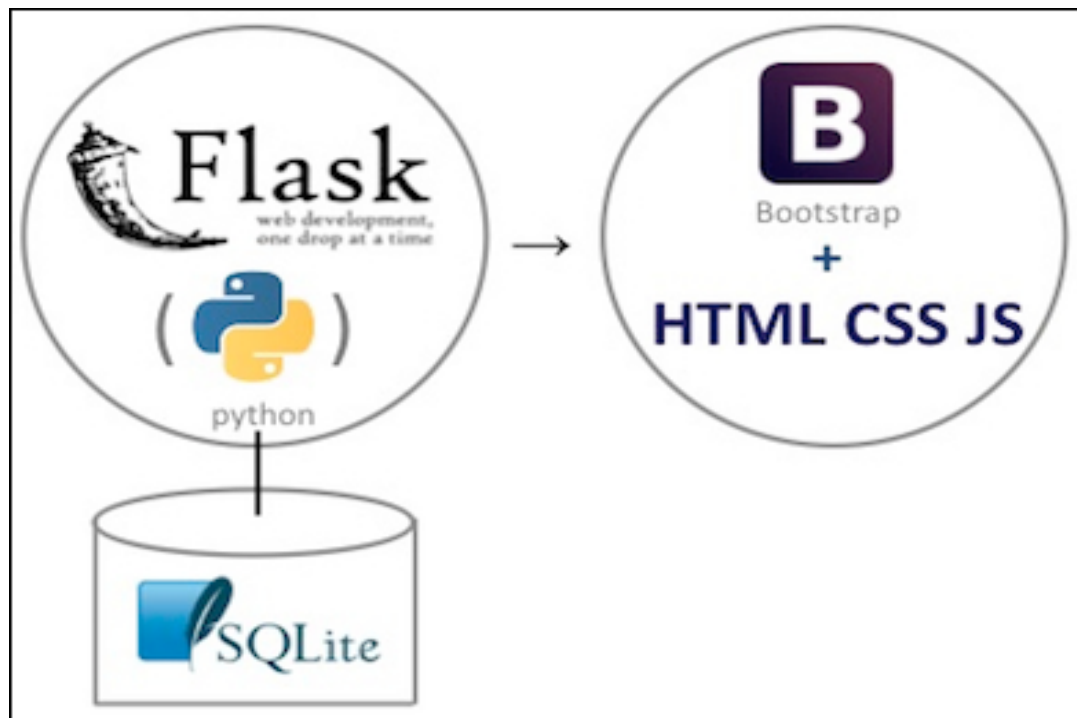


- Depuis la table chain on n'a utilisé que les colonnes suivantes :
 - La colonne structure_id.
 - La colonne chain_name.
- Depuis la table **nucleotide** on a utilisé les colonnes suivantes :
 - La colonne Chain_id.
 - La colonne index_chain.
 - La colonne paired.
 - La colonne paired_LW.



3. Langage et bibliothèques utilisés

- Pour la réalisation du projet on a choisi d'utiliser le langage de programmation python vu la flexibilité qu'il offre et le grand nombre des ressources existantes sur le web.
- Pour l'extraction des données depuis la base de données SQLITE on a utilisé le package sqlite3 de python, on a aussi utilisé le package Pandas pour manipuler les fichiers d'extension csv.
- Pour la représentation des graphes on a utilisé la bibliothèque Networkx.
- On a aussi utilisé la bibliothèque Pyvis afin de présenter le graph dans une page web.
- On a utilisé le Framework Flask qui est framework web qui nous a permis de rendre toute l'application web afin qu'elle soit plus simple à utiliser.


















pandas


PyViz


NetworkX
 python

4. Architecture du projet

 dao	adding dictionary of paire types to nucleotides manager	2 weeks ago
 managers	change motif from carnaval	1 hour ago
 models	constructing subgraph algorithm	1 week ago
 templates	architecture 3tiers + web flask	2 weeks ago
 .gitignore	adding __pycache__ to git ignore and remove .idea and __pycache__ fo...	2 weeks ago
 app.py	adding dictionary of paire types to nucleotides manager	2 weeks ago
 chains.csv	add .idea to gitignore	2 weeks ago
 gameofthrones.html	functional colored pairs	2 weeks ago
 main.py	adding a manager	2 hours ago
 networkRNA.html	mergin with fix_bug_managers	2 hours ago
 nucleotides.csv	mergin with fix_bug_managers	2 hours ago
 nx.html	colred relations	2 weeks ago
 plot_edge_colormap.py	colred relations	2 weeks ago
 sample.csv	constructing subgraph algorithm	1 week ago

- On a respecté l'architecture 3tiers pour la réalisation de notre projet :
 - La couche dao qui permet d'interagir avec la base de données et d'exécuter les requêtes SQL.
 - La couche Business « Managers » qui contient tout le code et le traitement qui permet de manipuler les données.
 - La couche présentation « Templates » qui contient les différentes page web de notre application.

5. Résultats

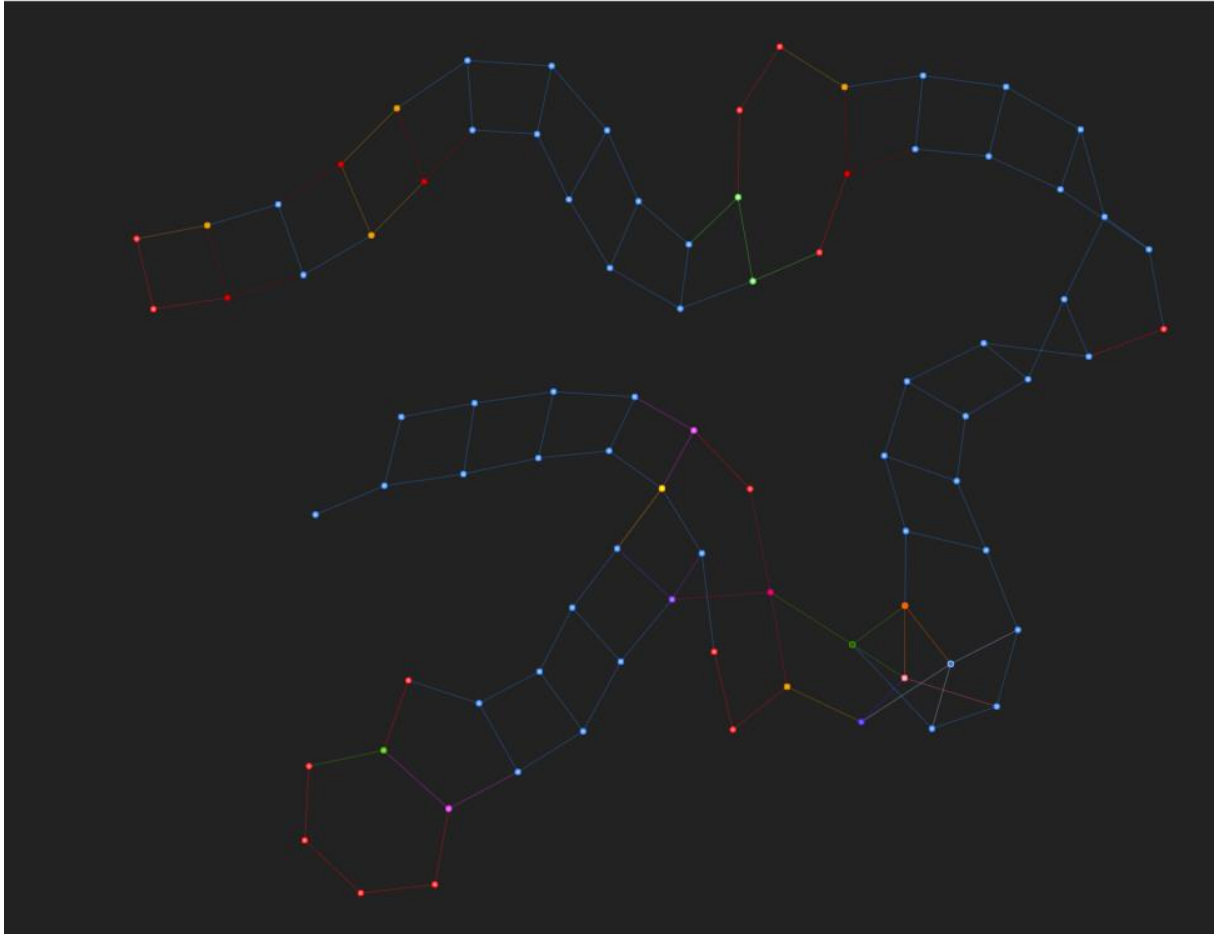
- Pour tester le fonctionnement de notre application on a testé quelque motif depuis carnaval. En effet nous avons réalisé notre propre algorithme récursif qui nous permet la détection des sous graphes et on a obtenu les résultats suivants :
- Liste des chaines des nucléotides existantes :

List of chains

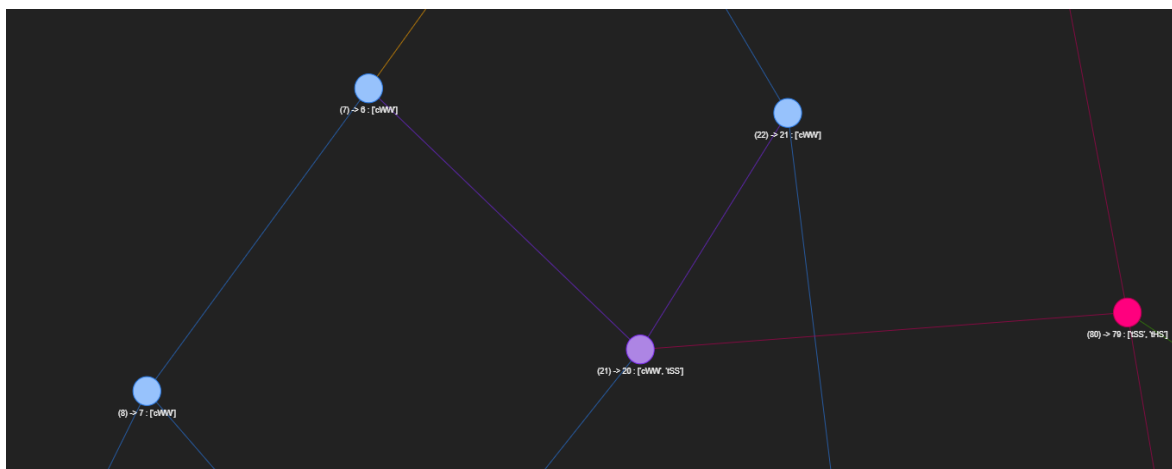
Chain_ID	Chain_Name	Eq_Class	
1	D	NR_20.0_33389.1	View chain
2	1	NR_20.0_99632.7	View chain
3	D	NR_20.0_55114.1	View chain
4	D	NR_20.0_74018.1	View chain
5	1	NR_20.0_99632.7	View chain
6	AV	NR_20.0_22323.2	View chain
7	q3	NR_20.0_22323.2	View chain
8	4	NR_20.0_12494.1	View chain
9	5	NR_20.0_04887.1	View chain
10	3	NR_20.0_97575.1	View chain
11	Bv	NR_20.0_71445.1	View chain
12	W	NR_20.0_22323.2	View chain
13	BA	NR_20.0_22323.2	View chain

- La représentation graphique de la chaîne des nucléotides choisis.

Nucleotids



- On a ajouté les étiquettes sur les arrêts pour indiquer le numéro des nucléotides et les types de liaisons.



- Détection du motif dans la chaîne des nucléotides.

```

=====
[ ==> self.chain_id: 105 self.index_chain: 57 paired : ['48'] self.paired_type: ['tHS']
, ==> self.chain_id: 105 self.index_chain: 58 paired : ['47'] self.paired_type: ['cWW']
, ==> self.chain_id: 105 self.index_chain: 47 paired : ['58'] self.paired_type: ['cWW']
, ==> self.chain_id: 105 self.index_chain: 59 paired : ['46'] self.paired_type: ['cWW']
]
=====
[ ==> self.chain_id: 105 self.index_chain: 65 paired : ['39'] self.paired_type: ['tHS']
, ==> self.chain_id: 105 self.index_chain: 66 paired : ['38'] self.paired_type: ['cWW']
, ==> self.chain_id: 105 self.index_chain: 38 paired : ['66'] self.paired_type: ['cWW']
, ==> self.chain_id: 105 self.index_chain: 67 paired : ['37'] self.paired_type: ['cWW']
]
=====
[ ==> self.chain_id: 105 self.index_chain: 80 paired : ['21', '25'] self.paired_type: ['tSS', 'tHS']
, ==> self.chain_id: 105 self.index_chain: 21 paired : ['7', '80'] self.paired_type: ['cWW', 'tSS']
, ==> self.chain_id: 105 self.index_chain: 7 paired : ['21'] self.paired_type: ['cWW']
, ==> self.chain_id: 105 self.index_chain: 22 paired : ['6'] self.paired_type: ['cWW']
]
=====

```

6. Instructions d'installation

- Copier la base de données SQL dans le répertoire du projet.
- Accéder au projet via la ligne de commande.
- Installer les packages utilisées à l'aide du package manager de python « pip »:
 - pip install pandas sql3 flask networkx pyvis
- Pour lancer l'application exécuter la commande suivante :
 - python -m flask run

Conclusion

Ce projet nous a permis de mieux découvrir les problématiques de biologique en relation avec l'informatique et comment les résoudre.