КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ВИСОКИХ ТЕХНОЛОГІЙ

Завідувач кафедри супрамолекулярної хімії
проф. Рябухін Сергій Вікторович
Протокол №___ засідання кафедри
від «___» _____2023 р.

РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ СТАНДАРТИЗАЦІЇ КРИСТАЛОГРАФІЧНИХ СТРУКТУР МЕТАЛ-ОРГАНІЧНИХ КАРКАСІВ

Випускна кваліфікаційна робота магістра
студентки спеціальності 102 Хімія
ОП «Хемоінформатика»
Капелюхи Анни Олексіївни

Науковий керівник
доцент кафедри органічної хімії
хімічного факультету КНУ імені Тараса Шевченка
д.х.н. **Григоренко Олександр Олегович**

Оцінка захисту роботи
_____

Київ - 2023р.

АНОТАЦІЯ

Капелюха А.О. Розробка програмного забезпечення для стандартизації кристалографічних структур метал-органічних каркасів – Випускна кваліфікаційна робота магістра за спеціальністю 102 Хімія ОП «Хемоінформатика».

У ході роботи було розроблено програмне забезпечення для стандартизації кристалографічних структур метал-органічних каркасів (МОК) шляхом видалення розчинника. Алгоритм призначений для видалення вільних нейтральних молекул, вільних заряджених молекул (протийонів) та зв’язаних розчинників з підрахунком заряду, вилученого із системи.

Встановлено, що розроблена програма має низький відсоток помилок і є набагато більш ефективною і порівнянні з існуючими рішеннями, що було доведено шляхом валідації на експериментальних структурах.

Новий алгоритм вже активно використовується для автоматизації підготовки CIF файлів МОК до високопродуктивного розрахункового скринінгу та розробки баз даних стандартизованих структур з мінімізованою кількістю помилок, що будуть розміщені у вільному доступі.

Ключові слова: метал-органічні каркаси, очистка кристалографічних структур, бази даних МОК для високопродуктивного скринінгу.

ЗМІСТ

СПИСОК УМОВНИХ СКОРОЧЕНЬ

ASA	Гравіметрично доступна площа поверхні, gravimetric accessible surface area
CCDC	Кембриджський центр кристалографічних даних, Cambridge Crystallographic Data Centre
CIF	Файл кристалографічної інформації, crystallographic information file
CoRE MOF	Підготовані до розрахунків, експериментальні метал-органічні каркаси, Computation-Ready, Experimental Metal−Organic Frameworks
CSD	Кембриджська база структурних даних, Chembridge Structural Database
iVBS	Ідеалізована сума валентності зв’язків, idealized valence bond sum
LCD	Діаметр найбільшої порожнини, largest cavity diameter
МОК, MOF	Метал-органічний каркас, metal organic framework
PLD	Лімітуючий діаметр пор, pore limiting diameter

ВСТУП

Метал-органічні каркаси (МОК) це періодичні координаційні сполуки що складаються з неорганічних вузлів, металів, та органічних лігандів які можуть поєднуватися між собою в різних комбінаціях для утворення нових матеріалів з різноманітною топологією. Хімічний простір МОК дуже великий і налічує сотні тисяч експериментальних структур, які було успішно синтезовано та охарактеризовано.[1]

МОК є перспективними пористими матеріалами через їх високу питому поверхню, стабільність та можливість варіювати розмір пор. Матеріали цього типу є гарними сорбентами газів – гідрогену [2], метану [3], вуглекислого газу [4], [5]. Також вони використовуються як каталізатори в деяких реакціях, зокрема розщеплення води [6], селективного окиснення [7], фотохімічних перетвореннях [6]. Окрім цього МОК знаходять своє місце в розробці лікарських засобів – мають потенціал застосування для доставки активних сполук в живих організмах завдяки ефективній адсорбції та біосумісності [8]–[10]. У зондуванні їх використовують для виявлення різних аналітів, таких як важкі метали [11], вибухові речовини [12] і летючі органічні сполуки [13]. МОК також досліджуються для використання в електронних пристроях, таких як датчики [14] і транзистори [15].

Такий широкий спектр застосувань та приклади промислового застосування є причиною активних дослідження цих матеріалів. Для цього використовується високопродуктивний розрахунковий скринінг, який робить можливою відносно швидку оцінку сотень тисяч структур, що не було б можливим практичними методами. [16]

Успішність скринінгу матеріалів розрахунковими методами напряму залежить від якості оцифрованої структури, адже у випадку МОК навіть незначні відхилення можуть істотно впливати на властивості, наприклад адсорбцію газів. Ілюстрацію зміни структурних параметрів МОК з вилученням розчинника показано на Рис.1.

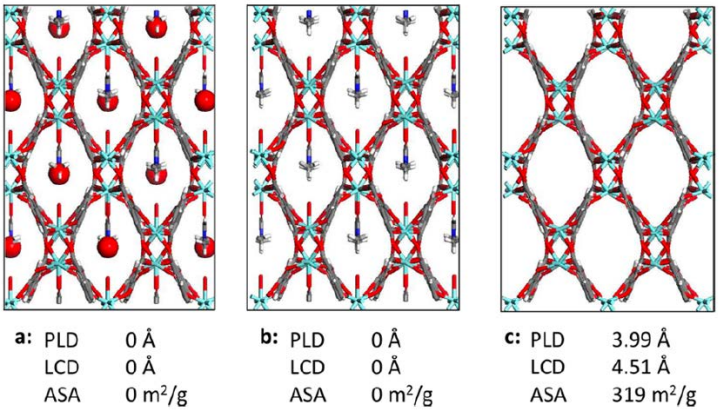


Рис. 1. Зміна розрахованих властивостей МОК залежно від структури [17]

Зазвичай для скринінгу використовують бази даних експериментальних кристалографічних структур МОК, а також бази теоретичних каркасів, які генеруються на їх основі. Найбільш популярними для скринінгу є бази CSD MOF Subset [18] та CoRE MOF 2019 [17].

Ці бази були сформовані з експериментальних структур шляхом підготовки шляхом вилучення розчинників. Проте, алгоритми які при цьому використовуються далекі від ідеалу, що призводить до великого відсотка помилкових МОК в цих базах даних. Також ці алгоритми абсолютно не враховують заряд вилучених фрагментів, що приводить до генерації структур з некоректним зарядом каркасу.

Зважаючи на це, нами було поставлено наступні цілі роботи:

- Аналіз типів помилок в структурах МОК бази даних CoRE MOF 2019;
- Ідентифікація недоліків існуючих алгоритмів вилучення розчинників, розробка нового програмного забезпечення для стандартизації МОК з урахуванням зарядів фрагментів;
- Валідація програми шляхом ручної перевірки правильності видалення розчинника;
- Порівняння ефективності розробленого методу з існуючими.

Розширені результати цієї роботи з додатковим аналізом баз даних експериментальних структур будуть опубліковані в профільному міжнародному журналі.

РОЗДІЛ І

ОГЛЯД ЛІТЕРАТУРИ

1.1 Зберігання та візуалізація кристалічних структур МОК

Експериментальні структури МОК зберігаються за допомогою файлів розширення *.cif (Crystallographic Information File). Вони містять інформацію про атоми та зв'язки між ними, кристалографічну симетрію, коефіцієнти термічного розширення, розміри комірок, дату створення, умови росту кристалу тощо. Файли CIF є стандартом обміну даними для кристалографічних структур і використовуються в багатьох програмах для моделювання та дослідження МОК. [19]

Форматування файлів цього типу може відрізнятися залежно від джерела з якого він походить або програм, яким він редагувався, проте він обов’язково має містити інформацію про якісний та кількісний склад кристалографічної структури, координати атомів та тип симетрії. Це дає можливість програмам-редакторам відтворити 3Д структуру для можливості подальшої роботи з нею.

Серед програм-редакторів найбільш розповсюдженими є наступні:

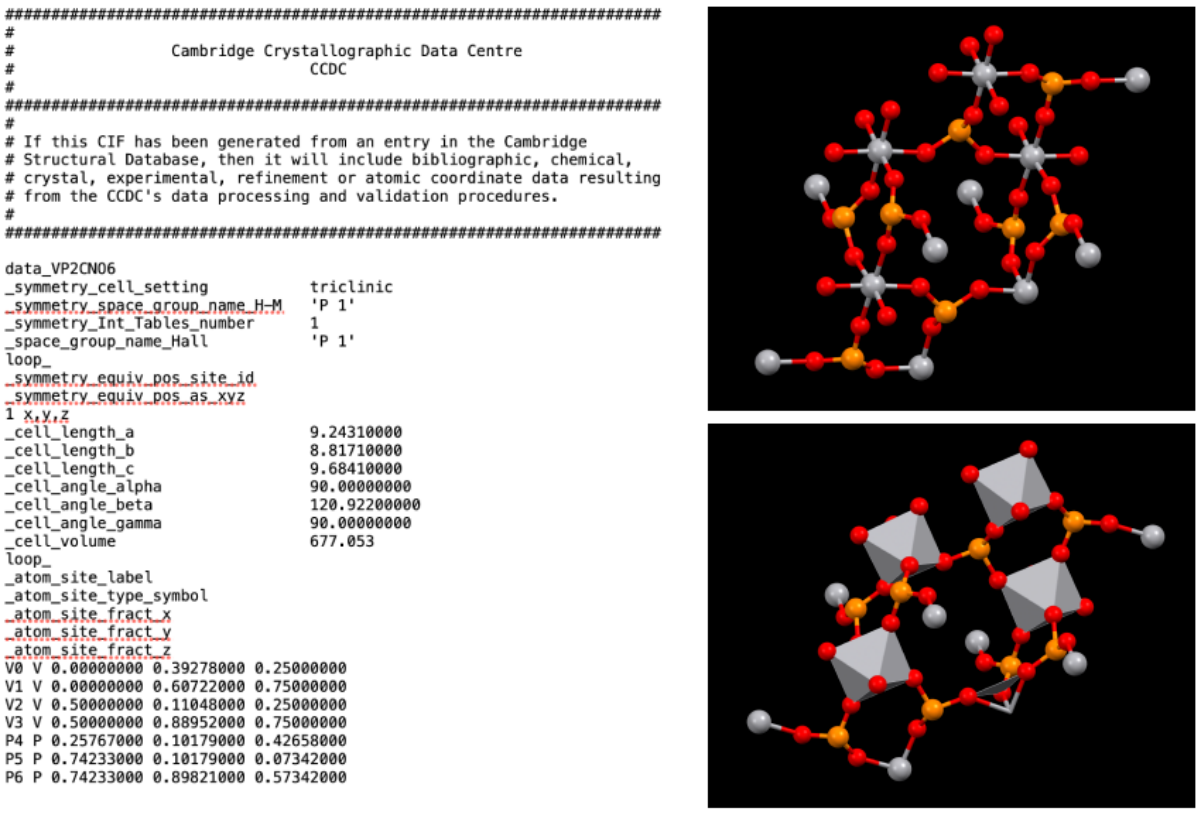
- Mercury. Це популярна програма для візуалізації та редагування кристалічних структур. Вона була розроблена Cambridge Crystallographic Data Centre (CCDC) та доступна для безкоштовного скачування. [20]
- Materials Studio. Комплексний набір програмних засобів для моделювання та імітації матеріалів. Він містить модуль під назвою "CIFEdit", який дозволяє користувачам редагувати CIF-файли МОК та інших матеріалів. [21]
- Vesta. Програма з відкритим кодом для візуалізації та аналізу кристалічних структур. Він містить редактор CIF, який дозволяє

користувачам редагувати позиції атомів та інші параметри файлів CIF. [22]

- Olex2. Зручна програма для аналізу кристалічних структур. Також містить редактор CIF, який дозволяє користувачам редагувати структуру, симетрію та інші параметри файлів CIF. [23]
- Avogadro. Популярне програмне забезпечення з відкритим кодом для молекулярного моделювання та візуалізації. Його також можна використовувати для редагування CIF-файлів МОК. [24]

Окрім цього, редагувати CIF файл можна напряму в текстовому редакторі, проте це не найбільш зручний спосіб для людини через відсутність графічної репрезентації.

Приклад вигляду CIF файлу відкритого в текстовому редакторі, а також у програмі Mercury зображені на Рис.2.



```
##################################################################
#
#                 Cambridge Crystallographic Data Centre
#                                 CCDC
#
##################################################################
#
# If this CIF has been generated from an entry in the Cambridge
# Structural Database, then it will include bibliographic, chemical,
# crystal, experimental, refinement or atomic coordinate data resulting
# from the CCDC's data processing and validation procedures.
#
##################################################################

data_VP2CNO6
_symmetry_cell_setting            triclinic
_symmetry_space_group_name_H-M    'P 1'
_symmetry_Int_Tables_number       1
_space_group_name_Hall            'P 1'
loop_
_symmetry_equiv_pos_site_id
_symmetry_equiv_pos_as_xyz
1 x,y,z
_cell_length_a                    9.24310000
_cell_length_b                    8.81710000
_cell_length_c                    9.68410000
_cell_angle_alpha                 90.00000000
_cell_angle_beta                  120.92200000
_cell_angle_gamma                 90.00000000
_cell_volume                      677.053
loop_
_atom_site_label
_atom_site_type_symbol
_atom_site_fract_x
_atom_site_fract_y
_atom_site_fract_z
V0 V 0.00000000 0.39278000 0.25000000
V1 V 0.00000000 0.60722000 0.75000000
V2 V 0.50000000 0.11048000 0.25000000
V3 V 0.50000000 0.88952000 0.75000000
P4 P 0.25767000 0.10179000 0.42658000
P5 P 0.74233000 0.10179000 0.07342000
P6 P 0.74233000 0.89821000 0.57342000
```

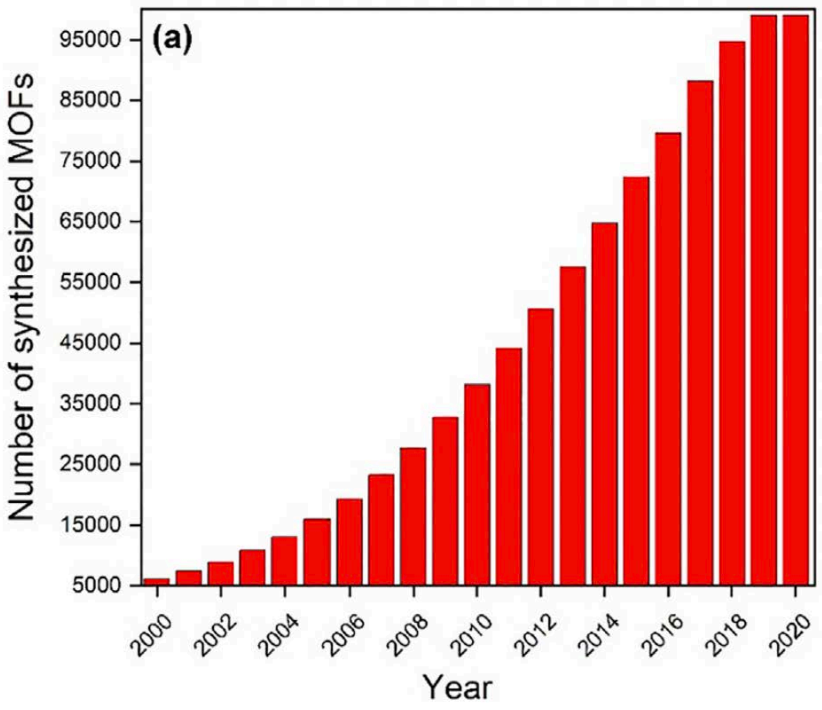
Рис.2. Зображення CIF файлу з рефкодом AHOJUC в текстовому редакторі та програмі Mercury

1.2 Найбільша база даних експериментальних структур - Cambridge Structural Database (CSD)

Кембриджська база структурних даних (Cambridge Structural Database, CSD) — це база даних кристалічних структур, яка підтримується Кембриджським центром кристалографічних даних (CCDC).

CSD була створена наприкінці 1960-х років і з тих пір перетворилася на важливий ресурс для дослідників у низці галузей, включаючи хімію, матеріалознавство та розробку ліків. Вона постійно поповнюється новими кристалічними структурами по мірі того як вони з’являються в наукових публікаціях. Це забезпечує актуальність цієї бази даних та робить її незамінним ресурсом для дослідників в цій галузі. [25]

Це також одна з найбільших і найповніших баз даних кристалічних структур у світі, яка містить понад 1,2 міліони структур за інформацією з офіційного сайту. [26] З них біля 100 тисяч є метал-органічними каркасами, кількість яких кожного року збільшується. Графік зміни кількості МОК в цій базі



зображений на Рис.3. [27]

Рис.3. Графік зміни кількості структур МОК в базі CSD [27]

Дані в CSD підбираються та анотуються командою експертів і доступні для підписників через низку програмних засобів та веб-інтерфейсів. Також варто підмітити, що ці структури є нередагованими, тобто вони містять розчинники, протийони та інші можливі артефакти. Саме через ці артефакти дані структури не підходять для високопродуктивного розрахункового скринінгу, адже не відповідають реальній картині «активованої» структури метал-органічного каркасу. [17]

1.3 Бази даних МОК зі структурами, підготованими до високопродуктивного скринінгу.

Високопродуктивний розрахунковий скринінг дає нам можливість досліджувати значно більший хімічний простір МОК для ідентифікації перспективних структур-кандидатів з визначеними властивостями. [16] Наприклад, структури з вищезгаданої бази даних CSD були використані для такого скринінгу з успішною ідентифікацією матеріалів для застосування в розділенні легких (CO_2, CH_4, H_2) [28], благородних газів [29], а також систем CO_2/N_2 [30].

Подібні дослідження потребують баз даних структур, спеціально підготованих до розрахунків, так званих “computation-ready databases”.

Теоретично вони мають містити «активовані структури» які відповідають метал-органічному каркасу з вилученими розчинниками та виправленими структурними помилками. [17]

1.3.1 CoRE MOF Database (2014)

Першою базою даних опублікованою в вільному доступі, розробники якої спробували закрити ці потреби, стала база **CoRE MOF Database** (CoRE MOF - Computation-Ready, Experimental Metal−Organic Frameworks – Підготовані до розрахунків, експериментальні метал-органічні каркаси). Вона була опублікована в 2014 році та налічувала 5109 експериментальних структур. [17] Ціллю розробників було спростити процес розрахункового скринінгу для інших дослідників цієї галузі.

Процес підготовки структур в базі CoRE зображений на Рис.4. За відправну точку була взята база даних CSD версії 5.35, яка налічувала структури, що були розміщені там до лютого 2014 року, сумарно більше 600 тисяч файлів.

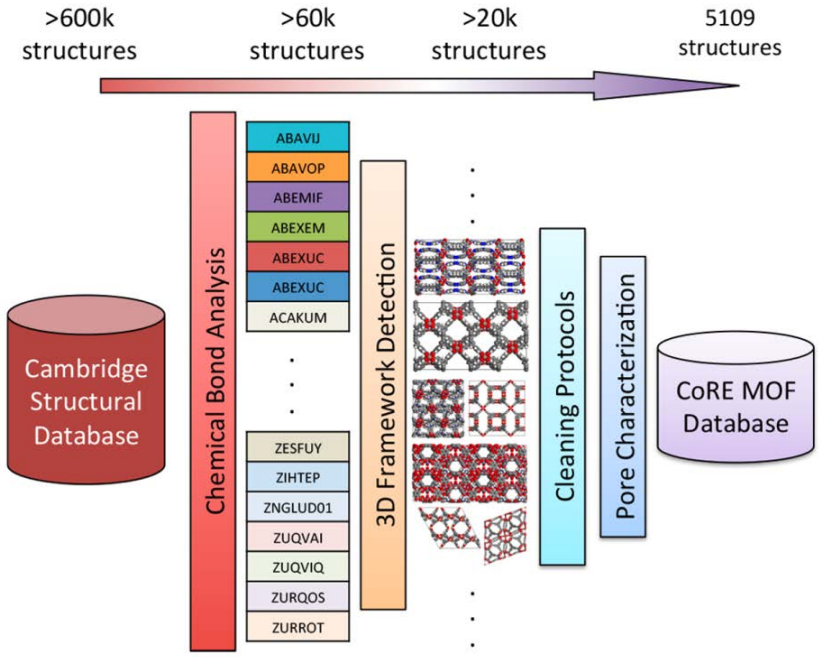


Рис. 4. Схема процесу підготовки структур МОК, стандартизованих для розрахунків [17]

Для фільтрації МОК та їх стандартизації було застосовано наступні кроки:

- Пошук структур, які містять більше одного зв’язку між металами та елементами O, N, B, P, S, та C. Ці атоми в свою чергу мають бути додатково зв’язані з C, N, P, чи S – близько 60 000 структур.
- Вибір матеріалів з 3Д структурою – близько 20 000 структур.
- Протоколи очистки – підрахунок зарядів протийонів, видалення невпорядкованих атомів, розчинників, ручне редагування – фінальні 5109 структур.

Додатково для кожного метал-органічного каркасу була проведена оптимізація геометрії за допомогою модуля Materials Studio Forcite. [17]

Ця база даних деякий час була єдиним доступним рішенням, тому дуже активно використовувалась для скринінгів різного типу, велика частина яких фокусується на дослідженнях адсорбційних властивостей цих матеріалів. До прикладу можна навести дослідження зі зберігання метану [31], очищення природного газу [32], захоплення CO_2 [33], розділення Xe/Kr [34], низькомолекулярних вуглеводнів [35] тощо. На момент написання даної роботи стаття має 443 цитування.

1.3.2 CSD Subset (2017)

Наступним після CoRE MOF було опубліковано базу **CSD Subset** (Cambridge Structural Database Subset), перша версія якого налічувала 69666 стандартизованих структур МОК. [18] На відміну від попередньої, ця база регулярно оновлюється і наразі налічує 114 373 записи (версія CSD 5.43, дані з офіційного сайту CCDC) [36]

Алгоритм створення цієї бази значно відрізняється та містить в собі такі етапи:

- Вибірка структур з CSD за текстовим пошуком за допомогою ConQuest – ключова програма для пошуку та отримання інформації з CSD [37]. Текстові запити пошуків ітераційно модифікувались для задоволення 7

ключових критеріїв структур (Рис. 5), а також щоб захопити усі структури, які входили до CoRE 2014.

- Стандартизація структур за допомогою скриптів для вилучення розчинників.

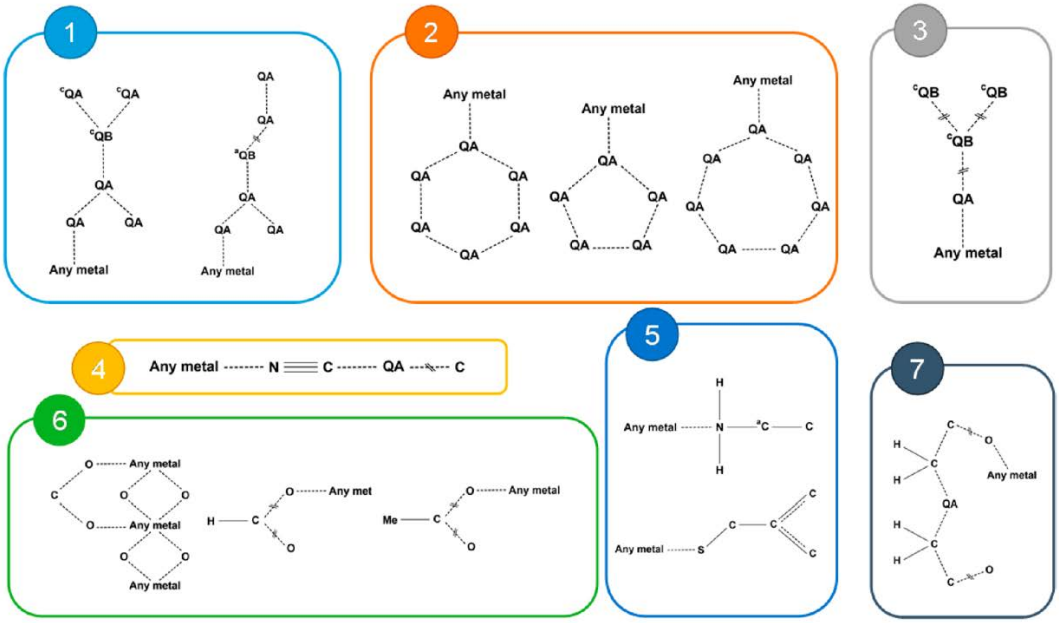


Рис. 5. Критерії відбору МОК для дизайну CSD MOF Subset

Бачимо, що алгоритм підготовки даної бази є повністю автоматизованим та не включає в себе ручне редагування файлів. Оригінальна публікація має більше 580 цитувань, що свідчить про активне використання цієї бази для різноманітних скринінгів.

1.3.3 CoRE MOF 2019

Після успіху попередньої бази даних та зі значним поповненням CSD, в 2019 році CoRE було розширено до більше 14 000 сполук. Ця вибірка була створена за протоколом аналогічним до CoRE-2014.

База CoRE MOF 2019 стала золотим стандартом у сфері дослідження властивостей метал-органічних каркасів, має більше 250 цитувань та активно використовується для високопродуктивного розрахункового скринінгу для різних потреб, найбільшу частину цих досліджень складає моделювання адсорбції газів. [38]

1.4 Огляд алгоритмів стандартизації МОК

Для зручності в подальшому в цій роботі буде згадуватися лише база даних CoRE MOF 2019, адже всі структури версії 2014 року входять до неї, а також вони мають спільний протокол очистки.

Загалом алгоритми стандартизації є алгоритмами вилучення розчинників, які мають забезпечувати наступне:

- Вилучення вільного розчинника. Під вільним розчинником маються на увазі нейтральні молекули, які не мають зв’язків з основним полімерним каркасом.
- Вилучення/ідентифікація протийонів, підрахунок заряду. Протийонами вважаються заряджені молекули, які не мають зв’язків з основним полімерним каркасом. Коректна ідентифікація та підрахунок заряду в даному випадку є надзвичайно важливим, адже властивості заряджених і незаряджених МОК істотно відрізняються. [39]
- Вилучення зв’язаного розчинника. Передбачає собою нейтральні молекули, які мають зв’язки з металами основного полімерного каркасу.

1.4.1 Протокол стандартизації CoRE MOF

На жаль, автори оригінальної статті не опублікували скрипти, за допомогою яких відбувалася очистка, тому ми можемо розглядати алгоритм стандартизації лише аналізуючи кроки, наведені в публікації.

Визначення заряджених протийонів

Для відділення нейтральних молекул розчинника від заряджених йонів аналізуються окремі компоненти структури. Під компонентами в даному випадку маються на увазі окремі молекули, які входять до складу кристалу, в тому числі й сам метал-органічний каркас.

Спочатку було використано модуль NeighbourList в Atomic Simulation Environment [40] для конструювання періодичної матриці суміжності для кожної структури. Два атоми вважались зв’язаними якщо відстань між ними менша за суму їх ковалентних радіусів + 0.4 Å. Далі ця матриця передавалась

до модуля connected components SciPy для ідентифікації зв’язаних компонентів в кожній молекулі. Зв’язані компоненти які відповідали структурам відомих йонів, які були записані в CSD, були ідентифіковані як протийони та залишені в кристалі з урахуванням відповідного заряду.

Вилучення вільних розчинників

На етапі вилучення розчинника всі зв’язані компоненти молекулярного графа були вилучені зі структури, окрім самого метал-органічного каркасу та протийонів. Сам метал-органічний каркас в даному випадку визначається як компонент кристалу з найбільшою молекулярною масою.

МОК в кристалах з періодичними фрагментами, що взаємно проникають один в одного (коли в одному кристалі може бути декілька окремих метал-органічних каркасів) визначаються шляхом фіксації кількості атомів найбільшого фрагмента N та зберіганні всіх фрагментів структури, кількість атомів в яких більша за 0.5N. [17]

Вилучення координованих розчинників

Вилучення координованих розчинників не є тривіальною задачею, адже вони зв’язані з основною періодичною структурою МОК та мають такі самі зв’язки як і ліганди, які є невід’ємною частиною каркасу.

Для цього авторами було розроблено наступний протокол:

- Молекула розділяється на частини шляхом розірвання зв’язку метал-оксиген.
- Якщо кількість окремих фрагментів в результаті такого розділення залишається незмінним, зв’язок відновлюється.
- У випадку збільшення кількості фрагментів, вся нова утворена молекула вважається розчинником і видаляється.
- Додатково прописане правило збереження ОН груп, що зв’язані з металами. [17]

Схематично процес видалення зв’язаних розчинників візуалізовано на прикладі CIF з рефкодом OFODET на Рис.6.

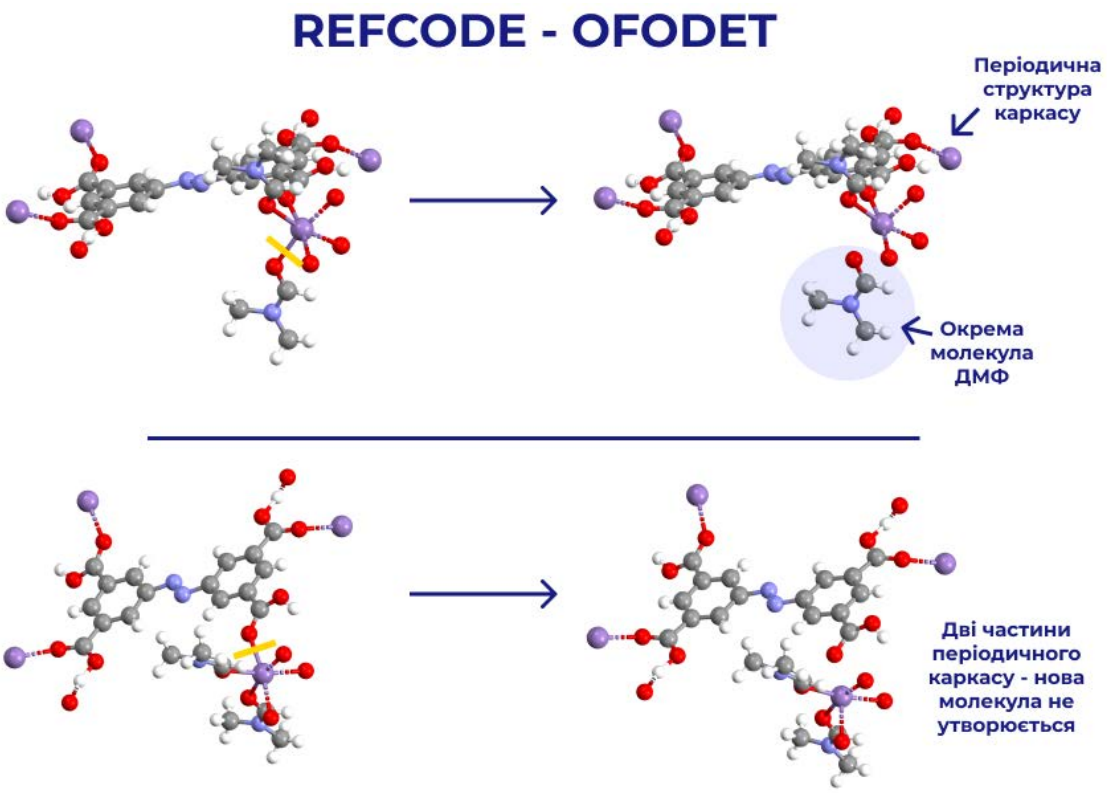


Рис. 6. Візуалізація видалення зв’язаного розчинника

Недоліками такого алгоритму вилучення розчинників є:

- Протийони, яких не має в базі CSD, не будуть визначені як заряджені структури. Мінус є суттєвим, тому що заряджені та незаряджені МОК мають дуже різні властивості і це суттєво впливатиме на результати скринінгу.
- За алгоритмом вилучення вільного розчинника МОК є структурою в файлі симетрії P1 з найбільшою кількістю атомів, що не завжди є правдою.
- У випадку структур з маленькою кількістю атомів в асиметричній одиниці правило зберігання вільних молекул з 0.5N атомів може також захоплювати розчинники.
- Алгоритм вилучення координованих розчинників не передбачає підрахунок заряду, а отже буде позбавлятися всіх монодентантних лігандів, незалежно від того заряджені вони чи ні.

- OH групи, що зв’язані з металами часто є молекулами води, в яких не вистачає атомів гідрогену. Всі такі молекули незалежно від їх природи залишаються в структурі.
- Алгоритм ніяк не намагається вирішити проблему термінальних атомів оксигену, які можуть бути як частиною структури, так і молекулами води з пропущеними атомами водню.

1.4.2 Протокол стандартизації CSD MOF Subset

Автори цієї бази даних опублікували скрипти за допомогою яких відбувалася очистка, тому ми маємо змогу детальніше розглянути їх в цій роботі. [18] Всі нові структури, які додаються до CSD MOF Subset проходять автоматичну підготовку за аналогічним алгоритмом.

Видалення вільних розчинників та протийонів

Вільні розчинники та протийони видаляються скриптом без будь-якого урахування заряду. Працює він на базі CSD Python API [41] та вилучає все, окрім молекули з найбільшою молекулярною масою за умови, що вона має полімерні зв’язки.

Видалення зв’язаних розчинників

Зв’язані розчинники вилучаються за наступним алгоритмом:

- В молекулі з найбільшою в кристалі молекулярною масою видаляються всі зв’язки у атомів металів.
- Атоми оксигену, які утворилися в результаті, видаляються, адже вони зазвичай відповідають молекулі води з пропущеними атомами гідрогену.
- Якщо структура утвореної молекули відповідає одному з розчинників, які є в списку CCDC, вона також видаляється, якщо вона не була зв’язана до цього з декількома металами.
- Фінальна структура МОК збирається назад шляхом відновлення решти зв’язків з металами. [18]

Огляд опублікованого коду

Алгоритм написано за допомогою мови програмування Python і для виконання йому необхідна CSD Python API, що доступна лише за платною ліцензією.

Скрипт нормалізує позначки на атомах кристалу перед роботою, прибирає зв'язки з металами за допомогою методу remove_bonds. Далі визначається список молекул для вилучення шляхом перевірки компонентів (компонент в даному випадку це молекулярний об'єкт який є частиною іншого молекулярного об'єкта). Шляхом ітерації списку з компонентами, що були ідентифіковані як розчинник, з оригінальної структури вилучаються відповідні атоми, які належать цим компонентам. Структура очищеного каркасу експортується у вигляді нового кристалу у форматі CIF.

```python
# Iterate over entries
for entry in io.EntryReader(args.input_file):
    if entry.has_3d_structure:
        # Ensure labels are unique
        mol = entry.molecule
        mol.normalise_labels()
        # Use a copy
        clone = mol.copy()
        # Remove all bonds containing a metal atom
        clone.remove_bonds(b for b in clone.bonds if any(a.is_metal for a in b.atoms))
        # Work out which components to remove
        to_remove = [
            c
            for c in clone.components
            if not has_metal(c) and (not is_multidentate(c, mol) or is_solvent(c))
            ]
        # Remove the atoms of selected components
        mol.remove_atoms(
            mol.atom(a.label) for c in to_remove for a in c.atoms
        )
        # Write the CIF
        entry.crystal.molecule = mol
        with io.CrystalWriter('%s/%s_stripped.cif' % (args.output_directory, entry.identifier)) as writer:
            writer.write(entry.crystal)
```

Недоліки алгоритму бази CSD MOF Subset:

- Повністю відсутнє будь-яке урахування заряду при видаленні вільних розчинників.
- Атоми оксигену, які видаляються автоматично, можуть бути не молекулами води, а групами типу Me=O та Me-OH, що досить часто

зустрічається в структурах МОК. В цьому випадку вони будуть видалені без попереднього аналізу.

- Зв’язані розчинники, яких немає в списку CCDC залишаться частиною структури.

1.5 Висновки з літературного огляду

Розглянувши проблему стандартизації структур для високопродуктивного розрахункового скринінгу метал-органічних каркасів, ми побачили, що існуючі алгоритми не можуть забезпечити їх ефективну підготовку. В результаті використання цих протоколів розчинники та протийони не видаляються в повній мірі, видаляються неправильно або без коректного урахування заряду системи. Це може істотно впливати на результати скринінгу, а отже дослідники отримуватимутьі некоректні результати, які навіть неможливо перевірити не проводячи експериментальну валідацію вибраних матеріалів.

Цей факт дуже знижує цінність теоретичних робіт, які використовували вищезгадані бази даних (а це більшість враховуючи їх високу популярність, загалом 3 оригінальні статті мають майже 1300 цитувань), оскільки якщо авторам не «пощастило» вибрати правильно оброблену структуру своїм топ-кандидатом, то з великою вірогідністю цей метал-органічний каркас матиме істотно інші властивості. Також це перешкоджає ідентифікації справді хороших матеріалів, які через неправильну структуру показують гірші результати.

Для ефективного дослідження великих баз даних МОК необхідна розробка алгоритму вилучення розчинників, який зможе забезпечити максимально гарні результати з мінімальним втручанням людини. Також він зможе стати частиною протоколу підготовки структур для їх розміщення в базах даних для високопродуктивного розрахункового скринінгу.

РОЗДІЛ ІІ

АНАЛІЗ ПОМИЛОК В БАЗАХ ДАНИХ ТА РОЗРОБКА НОВОГО АЛГОРИТМУ

Під час аналізу існуючих алгоритмів стандартизації баз даних було ідентифіковано ряд значних недоліків цих алгоритмів, тому нами було вирішено проаналізувати помилки в структурах метал-органічних каркасів, які входять до вищезгаданих баз. Аналіз відбувався з використанням ще неопублікованих даних, які були отримані в науковій групі професора Тома К. Ву відділу Хімії та Біомолекулярних Наук університету Оттави. В даній роботі подається лише аналіз бази CoRE MOF 2019, адже на момент її написання аналіз CSD MOF Subset не було завершено.

2.1 Аналіз помилок в структурах МОК в базі даних CoRE MOF 2019

На жаль, на даний момент не існує методу повної автоматизації визначення коректності структур в CIF файлах МОК, тому велика частина аналізу була проведена вручну з застосуванням додаткових алгоритмів, розроблених силами наукової групи професора Тома К. Ву (алгоритми на даному етапі не опубліковані в науковій літературі, проте готуються до публікації). Зокрема використовувалася програма MOSAEC, яка була створена для аналізу зарядів в метал-органічних каркасах та виявлення в них помилок.

База даних CoRE MOF 2019 складається з двох частин:

- CoRE MOF 2019 – FSR. FSR – Free Solvent Removed, в цьому сеті представлено структури МОК, в яких було вилучено тільки вільний розчинник (не зв’язаний з основним полімерним каркасом). Налічує 6008 структур.

- CoRE MOF 2019 – ASR. ASR – All Solvent Removed, структури в яких було вилучено всі розчинники: вільні, зв’язані та протийони. Налічує 10143 структури. [38]

Схематичне зображення складу цієї бази поміщено на Рис. 7.

Рис. 7. Схематичне зображення складу бази CoRE MOF 2019, адаптовано з [38]

Для аналізу структур була обрана наступна стратегія:

- За допомогою програми MOSAEC було визначено чи містять структури потенційні проблеми.
- Проведений аналіз результатів. Структури, позначені як GOOD, тобто хороші, були відкладені без подальшого аналізу, адже алгоритм є досить точним, тому похибка такої класифікації мінімальна.
- Структури позначені BAD були проаналізовані в ручному режимі, адже програма не може автоматично визначати в чому саме полягає проблема. Було виділено декілька основних типів помилок, які містилися у поганих структурах:

- Відсутність атомів гідрогену на лінкерах або атомах, зв’язаних з металами;
- Неправильний заряд полімерного каркасу;
- Повна відсутність лігандів або металів, які мають бути в структурі;
- Наявність невпорядкованих атомів.

Результати будуть розглянуті окремо по двом сетам, адже вони мають різний протокол створення.

CoRE MOF 2019-ASR

Статистика для сету, з якого було вилучено всі типи розчинників зображена на інфографіці на Рис.8.

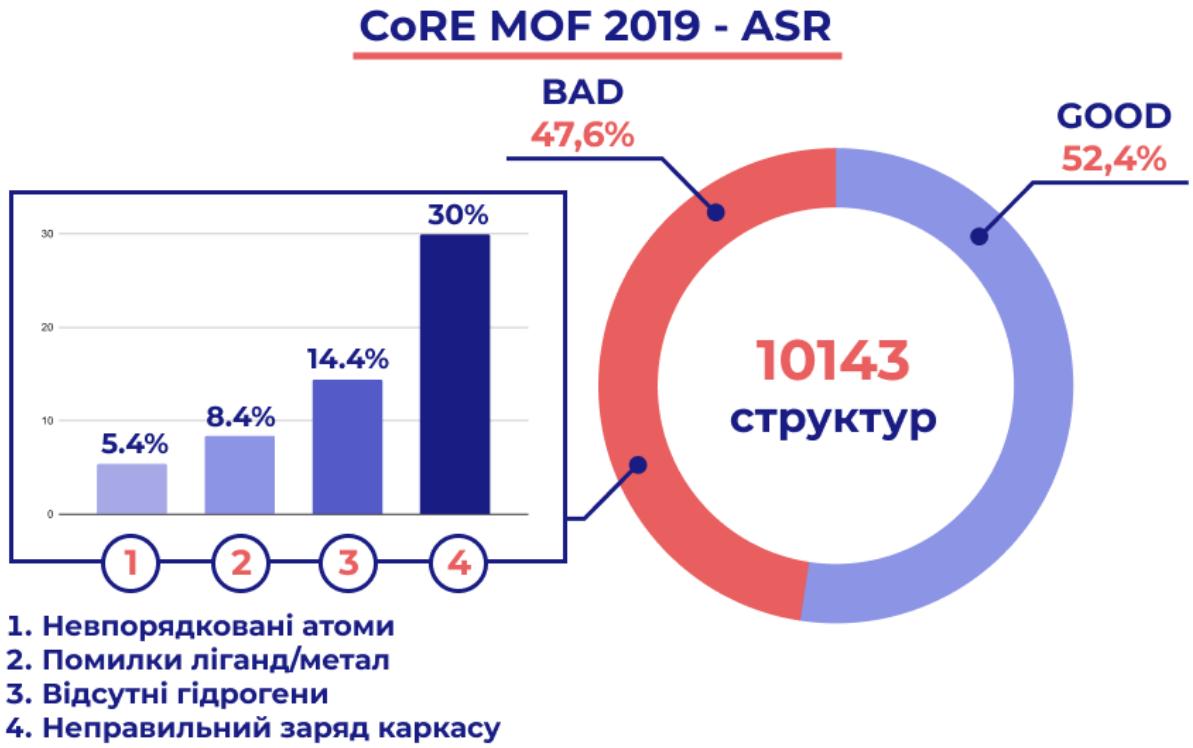


Рис. 8. Статистика помилок сету CoRE MOF 2019-ASR

Бачимо, що метал-органічні каркаси з неправильними структурами становлять 47,5% сету. Такий результат нас дуже здивував, адже ця база була неодноразово використана дослідниками для проведення скринінгів, а отже багато з них не дуже уважно підходять до даних з якими працюють. Також це

ще раз ставить під сумнів правильність суто теоретичних розрахункових досліджень.

Якщо ми детальніше розглянемо помилки, то побачимо, що найбільшу частку займає неправильний заряд системи (30%). Причиною цього може бути те, що алгоритм не розрізняє велику кількість протийонів, і вони в свою чергу вилучаються зі структури у вигляді нейтральних молекул.

14.4% структур мають проблеми з відсутністю атомів гідрогену на лінкерах або атомів зв'язаних з металами. Скоріше за все ці помилки не зв'язані з вилученням розчинника і є артефактами неправильного пост-процесингу структур після кристалографії.

Вилучення металів та лігандів є грубою помилкою, адже, як і неправильний заряд, це суттєво впливає на властивості матеріалу. Нами було ідентифіковано 8.4% структур з такими дефектами. Поясненням подібного явища є ще один недолік процесу вилучення розчинників – алгоритм не аналізує чи наявні метали в фрагментах, що вилучаються, а також не бере до уваги заряд цих фрагментів. Це означає, що будь-який монодентантний ліганд буде вилучено, навіть якщо він є зарядженим та має залишитися частиною структури.

Структури з невпорядкованими атомами є великою проблемою бази даних CSD, адже коли користувачі завантажують до неї структури, вони не проходять додаткову перевірку на коректність. Відповідно до цього в базу проникає велика кількість таких МОК, і такі помилки на даному етапі можуть бути виправлені лише вручну, адже для цього не існує ефективного програмного забезпечення. Структури з цим типом помилки складають 5.4%.

Досить схожу картину ми можемо побачити на Рис.9, де подано аналіз структур метал-органічних каркасів в CoRE MOF 2019-FSR (вилучено лише вільний розчинник – нейтральні молекули + протийони).

Примітка. Відсотки на графіках, що відповідають типу помилки в «поганих» структурах на Рис. 8 та 9 не дають в сумі 100%, адже один МОК

може містити в собі декілька типів помилок і вони всі взяті до уваги в цьому аналізі.

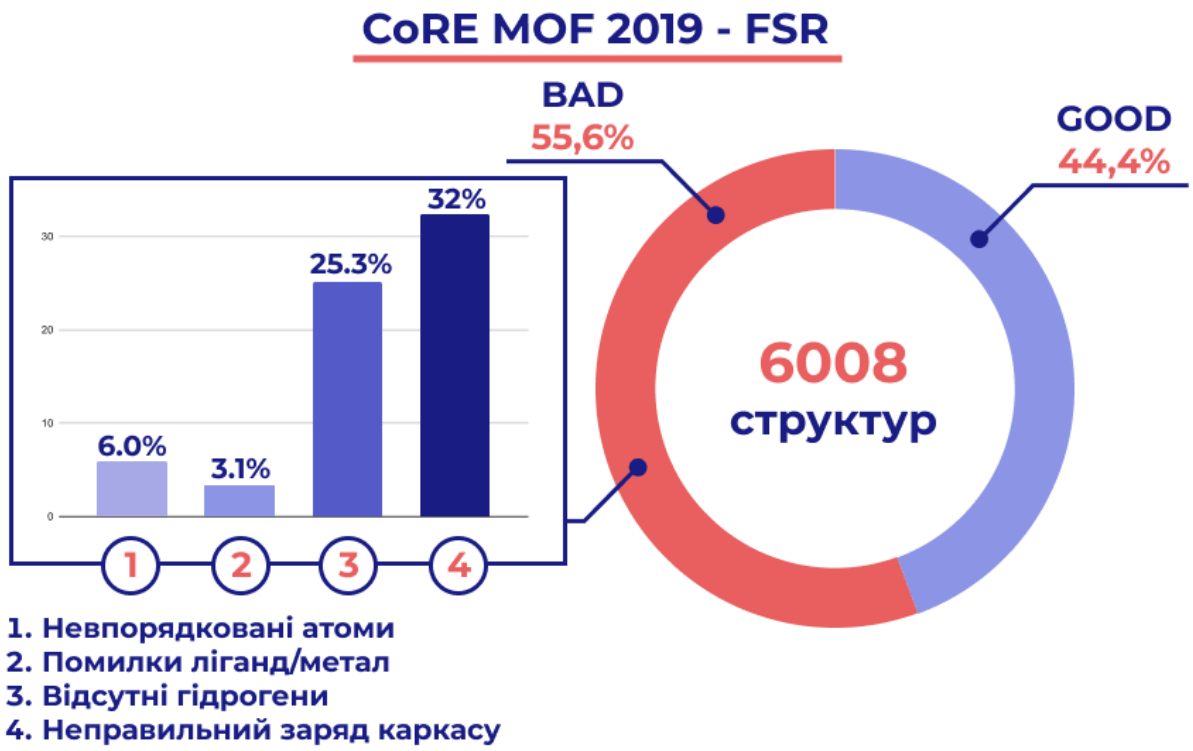


Рис.9. Статистика помилок сету CoRE MOF 2019-FSR

Для того, щоб порівняти склад помилок в цих сетах між собою розглянемо Рис.10.

Рис.10. Порівняння відсоткового складу помилок сетів ASR та FSR

До обох сетів входять лише теоретично пористі матеріали, тому кількість структур в ASR є вищою. Вилучення зв’язаного розчинника звільняє пори, тому розрахований параметр PLD (pore limiting diameter – діаметр, що обмежує пору) буде вищим, а саме за ним фільтрувалися структури. [38]

Ми не можемо порівнювати відсоткове співвідношення помилок в цих сетах, адже вони містять різну кількість структур, проте з графіку чітко видно, що кількість некоректних CIF файлів становить близько половини, що є неприпустимим для використання такої бази даних для їх дослідження чуттєвими розрахунковими методами.

З аналізу сетів бази даних CoRE MOF 2019 ми можемо зробити наступні висновки:

- Обидва сети бази мають високий відсоток помилкових структур, що складають близько половини загальної кількості. Цей факт робить дану базу непридатною для її використання в високоточних розрахункових методах.
- Значна частина цих помилок утворилась в результаті використання неефективного методу вилучення розчинника – 30.0 і 32.0 відсотки структур відповідно містять некоректний заряд, 8.4 та 3.4% містять помилкове видалення металів та лігандів (частково даний дефект може походити від процесу кристалографії).
- Відсутність атомів гідрогену також може частково бути результатом вилучення розчинників, проте це питання потребує додаткового аналізу. Одним з варіантів виправлення частини таких структур можуть стати автоматичні інструменти програми Mercury або нескладні скрипти написані з використанням функціоналу CSD Python API.
- Проблема невпорядкованих атомів на даний момент часу лишається невирішеною і її вирішення можливе лише шляхом ручного редагування.

2.2 Новий алгоритм – постановка задачі

Провівши аналіз помилок в базі даних CoRE MOF 2019 та проаналізувавши можливі недоліки двох широковживаних алгоритмів стандартизації МОК шляхом вилучення розчинника, ми прийшли до висновку, що існуючим протоколам не вистачає наступних можливостей:

- Універсальність. Обидва алгоритми частково покладаються на бази даних розчинників та протийонів CSD, що робить їх безсилими проти нетривіальних задач. В результаті цього велика кількість зв’язаних розчинників залишається частиною структури, що впливає на розраховану пористість матеріалу.
- Підрахунок заряду. Обидва алгоритми мають проблеми з зарядженими фрагментами, причому у випадку CSD MOF Subset ці заряди в принципі ігноруються повністю. База CoRE частково враховує заряди, проте лише обмеженої кількості протийонів. В результаті близько 30% структур мають неправильний заряд.
- Врахування термінальних атомів оксигену та ОН на металах. Термінальні атоми оксигену та ОН групи на металах можуть бути як координованою водою, якій бракує атомів водню, так і групами Me=O та Me-OH відповідно. Жоден з алгоритмів не бере це до уваги в повній мірі.
- Низький відсоток помилок. Новий алгоритм має враховувати більше мінорних особливостей в можливих артефактах файлів такого типу.

2.3 Схема роботи нового алгоритму для вилучення розчинника

Цей розділ буде повністю присвячено схемі роботи розробленого нами алгоритму. Схематично виконання коду програми зображено на Рис. 11.

Як вже було сказано раніше, програма працює виключно з файлами формату CIF, які є стандартом індустрії для зберігання електронних структур метал-органічних каркасів. Оскільки в цих файлах можлива варіація

форматування, спочатку кристал, що в ньому знаходиться, приводиться до стандартного формату.

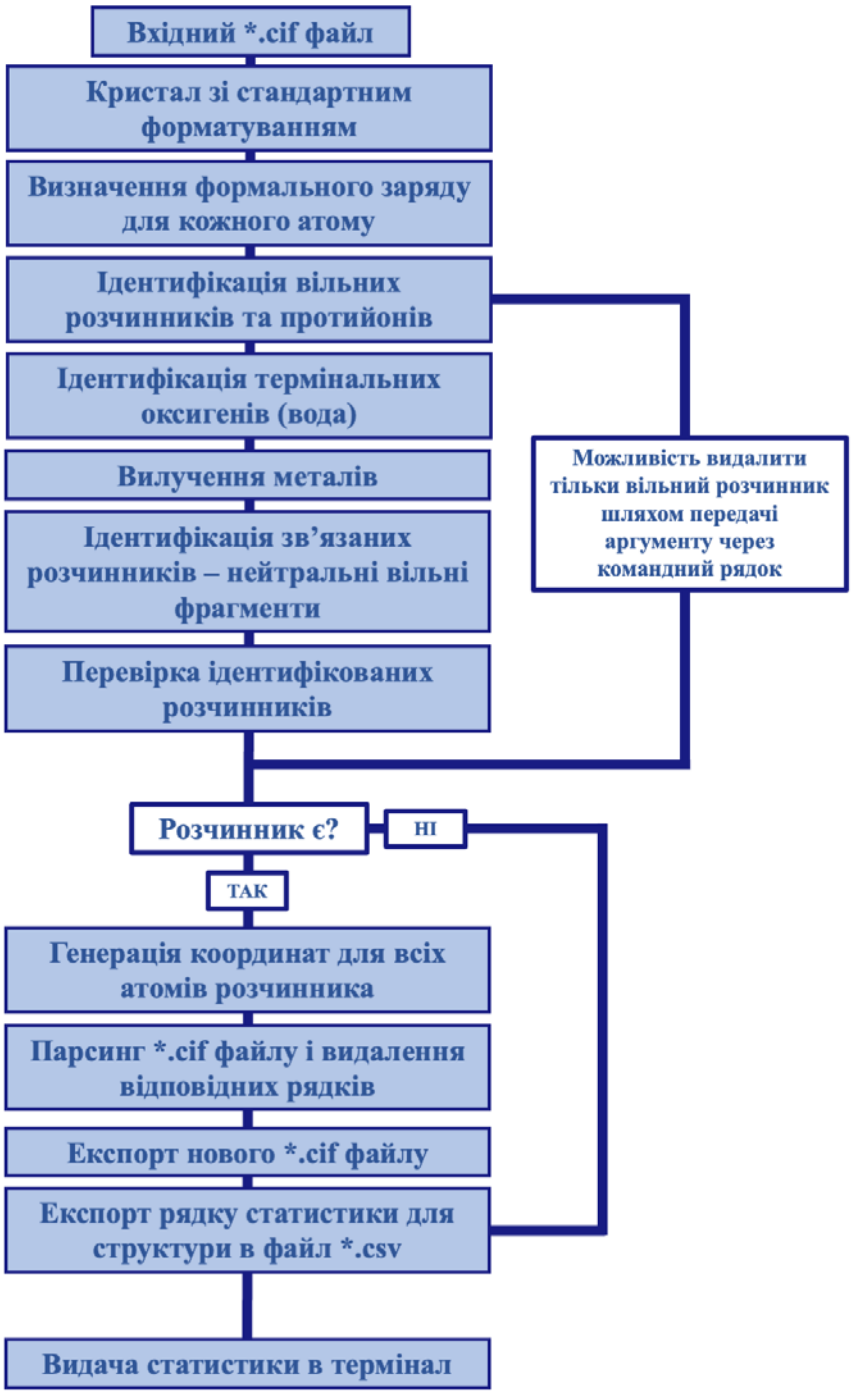


Рис.11. Схема виконання коду програми.

За допомогою частини функціоналу раніше згаданої програми MOSAEC відбувається підрахунок формального заряду для кожного атому.

Далі відбувається ідентифікація вільних розчинників та протийонів. Вони вилучаються зі структури, при цьому програма фіксує, які з частинок є зарядженими та зберігає цю інформацію. Процес можна завершити на цьому етапі, передавши в програму аргумент “--keep_bound” через командний рядок.

У випадку якщо потрібне вилучення всього розчинника, далі відбувається аналіз термінальних оксигенів. Для цього програма звертається до бази CSD і за рефкодом (ідентифікатор структури в базі) визначає чи мають в цій структурі бути термінальні атоми оксигену. Всі необхідні атоми вносяться до списку для вилучення.

Далі створюється копія молекулярного об’єкта кристалу та з нього вилучаються метали. В результаті молекулярний об’єкт ділиться на фрагменти, які аналізуються індивідуально. Шляхом аналізу формального заряду на атомах ідентифікуються нейтральні молекули. Список ідентифікованих розчинників передається в допоміжну функцію на перевірку. Критерії перевірки будуть детальніше розглянуті в наступному розділі.

Якщо розчинник у структурі не виявлено, то новий CIF файл не генерується і в CSV файл зі статистикою додається відповідний рядок. Якщо розчинник виявлено, для всіх ідентифікованих атомів генеруються координати x,y,z. Ці координати приводяться до стандартного формату. Далі програма створює копію вихідного CIF у вказаній папці та починає його сканувати як текстовий файл, вилучаючи рядки з координатами, які відповідають координатам атомів розчинника. Файл оновлюється новим текстовим наповненням.

Додатково статистика експортується у вигляді додаткового рядка до вищезгаданого CSV файлу. Для зручності статистика у спрощеному вигляді дублюється у термінал, щоб користувач міг зручно контролювати перебіг процесу.

Вилучення розчинника відбувається циклічно для всіх файлів у вказаній користувачем директорії. Для пришвидшення процесу та більш ефективних розрахунків передбачена паралелізація, яку можна налаштувати під можливості комп’ютера за допомогою аргументів командного рядка.

2.4 Структура коду та оформлення

Код програми опублікований у вільному доступі на GitHub за наступним посиланням: https://github.com/AnnaKapel/MOF_solvent_remover

Для зручності використання сторонніми людьми там подано детальну інформацію щодо процесу роботи з цим алгоритмом, доступні аргументи командного рядка та розшифровка колонок двох видів таблиць зі статистикою. Ця інформація поміщена у файл README.md, який містить в собі текст специфічного форматування, який платформа перетворює в структуровану інструкцію.

Вигляд репозиторію показаний на Рис. 12.

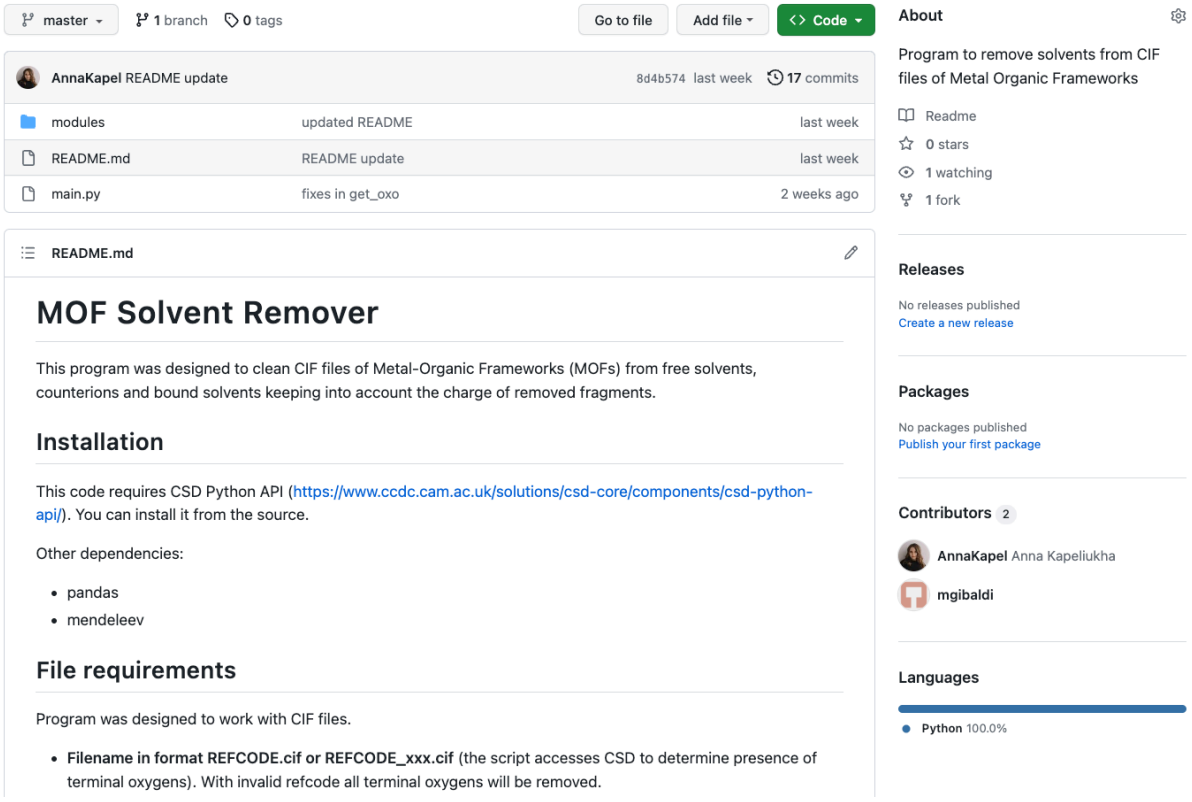


Рис. 12. Вигляд GitHub репозиторію

Сама програма складається з 5 окремих модулів та основного файлу main.py, де прописаний основний код виконання. Модулі складаються з функцій, які було розділено в залежності від того, до якого процесу відноситься їх дія. Додатковий файл __init__.py потрібен для того, щоб ми могли імпортувати функції з модулів в основне тіло коду. Структура програми та назви окремих модулів показані на Рис. 13.

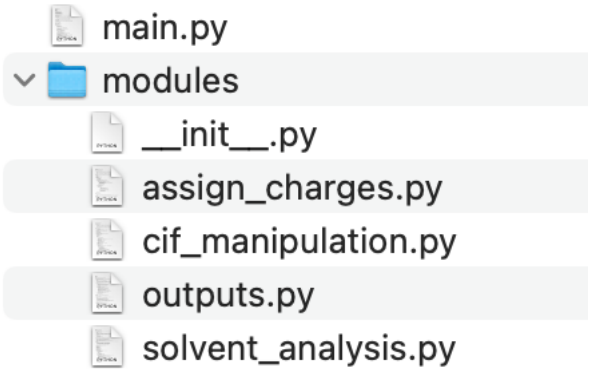


Рис. 13. Структура програми

2.5 Детальний розгляд роботи програми та коду

Цей розділ буде повністю присвячений детальному розгляду алгоритму роботи розробленої нами програми.

2.5.1 Огляд використаних інструментів

Для написання програми було вирішено обрати мову програмування Python, адже вона активно використовується для розробки програмного забезпечення в сфері хімічних досліджень. Не зважаючи на існування новіших версій пітону, використовувалась версія 3.7.*, адже саме для неї написана ключова бібліотека, яка використовувалась в коді програми. Таким чином ми забезпечуємо стабільну роботу алгоритму.

Сторонні бібліотеки, які використовуватись:

- CSD Python API. Інтерфейс програмування на мові Python для взаємодії з Cambridge Structural Database. Дозволяє здійснювати пошук, вибірку, взаємодію та аналіз кристалічних структур.
- Mendeleev. Дозволяє здійснювати обчислення хімічних властивостей елементів та їх сполук.

В даній роботі елементи коду не будуть подані повністю, тому що він є досить об’ємним. Для пояснення окремих процесів ми будемо наводити його частини. Повний код можна подивитися в репозиторії на GitHub за раніше вказаним посиланням.

Для зручності користування та можливості роботи з кодом програми сторонніми розробниками, усі функції модулів буди прокоментовані відповідно до загальноприйнятих правил англійською мовою. Дода

```
def get_rAON_atomlabels(rAON_old):
    """
    Creating a dictionary of rAON, where each value corresponds to atom
    label instead of an atom object

    Parameters:
        rAON_old (dict): dictionary atom: oxidation contribution for all atoms in the MOF.
    Returns:
        rAON_atomlabels (dict): dictionary atom label: oxidation contribution.
    """
```

2.5.2 Аргументи командного рядка та запуск програми

Даний алгоритм було розроблено для роботи з ним виключно через командний рядок без графічного інтерфейсу, оскільки цільова аудиторія для його використання це люди, які працюють з великими масивами даних, часто з залученням розрахункових кластерів.

Для запуску програми необхідно перейти в папку її установки та запустити **main.py**, передавши в нього всі необхідні аргументи.

```
cd <path_to_installation_folder>
python main.py <arguments>
```

Аргументи, які було вирішено додати та їх функціонал детальніше описано в Таблиці 1.

Аргумент	Стандартне значення	Пояснення
--data_path	папка з файлом main.py	Можливість задати шлях до папки, в якій розташовані вихідні CIF файли
--export_path	папка з файлом main.py	Можливість задати шлях до папки, в яку будуть експортуватись результати
--n_processes	4	Кількість процесів
-v, --verbose	False	Можливість вимкнути вивід статистики в командний рядок

--keep_bound	False	Можливість вилучення лише вільного розчинника та протийонів, при цьому зв’язаний залишається в структурі
--keep_oxo	False	Можливість залишити всі термінальні оксигени в структурах

Таблиця 1. Доступні аргументи командного рядка

2.5.3. Паралелізація процесу

Оскільки програма розрахована на роботу з великими масивами даних, нами було передбачено можливість паралельного виконання процесів. Для цього потрібно передати аргумент --n_processes через командний рядок відповідно до кількості CPU, які користувач хоче використати під виконання цього алгоритму. Стандартне значення 4.

Паралелізація була реалізована за допомогою вбудованого модуля multiprocessing, за допомогою метода Pool.imap() ми створюємо ітератор, який застосовує функцію worker до кожного з об’єктів в ітераторі. В нашому випадку об’єкт – це назва файлу CIF у вказаній папці, а worker – це функція, в яку поміщено весь функціонал, який треба застосувати до цього файлу. В результаті програма може обробляти декілька файлів одночасно.

```
pool = Pool(processes = int(args.n_processes))

for res in pool.imap(worker, files):
    if res is not None:
        export_res(res, keep_bound, output_dir)
```

Також до функції worker було додано конструкцію try-except, яка забезпечує продовження виконання програми у випадку виникнення помилок. Помилкою може бути, наприклад, некоректний CIF файл, що не містить координат, атомів тощо. У такому випадку користувач буде повідомлений про помилку через командний рядок.

2.5.4. Читання файлу та підготовка кристалічної структури

Оскільки алгоритм виконується паралельно, назви файлів у вказаній користувачем директорії (за допомогою --data_path) записуються у список, який далі використовується для почергового їх відкривання. Папка може містити файли і іншого типу, тому для виділення саме CIF прописана додаткова умова.

Для обробки файлу використовується функція **readentry()** модуля **cif_manipulation**. Яка приймає назву файлу, читає його за допомогою CrystalReader() CSD Python API (бібліотека ccdc за назвою імпорту). Це потрібно лише для того, щоб можна було виділити емпіричну формулу кристалу.

Інформація про кристал отримується шляхом парсингу CIF як текстового файлу, таким чином ми можемо отримати правильну інформацію про ідентифікатори атомів, а також прибрати дублікати, які є частими артефактами в структурах. Додатково ми також оновлюємо нумерацію атомів, щоб уникнути ідентифікаторів, що повторюються та можуть викликати проблеми в подальшому. Нова нумерація починається з 1 для кожного типу атому.

З нового відредагованого текстового файлу, який зберігається в оперативній пам’яті та не експортується, генерується новий кристалічний об’єкт (ccdc.crystal.Crystal). Він представляє собою набір атомів, тому додатково в ньому визначаються зв’язки за допомогою методу assign_bonds(). Цей об’єкт зі зв’язками повертається функцією.

2.5.5. Визначення формального заряду атомів

Для визначення формального заряду на атомах використовується функціонал програми MOSAEC, яка була розроблена в групі професора Тома К. Ву Університету Отави. Ця програма поки що не опублікована у профільних виданнях, проте вона викладена у вільному доступі за наступним посиланням: https://github.com/uowoolab/MOSAEC

MOSAEC було розроблено для присвоєння формального заряду металам в каркасі, шляхом аналізу цього заряду програма робить висновок про правильність структури. Це в майбутньому дозволить аналізувати великі кількості файлів МОК для ідентифікації потенційно проблемних структур. На момент написання цієї роботи алгоритм повністю протестовано, підтверджено низький рівень похибки аналізу за допомогою нього, а також з його допомогою проводиться аналіз доступних баз даних метал-органічних каркасів.

В нашій роботі став корисним фрагмент цієї програми, який підраховує формальні заряди на атомах користуючись модифікованою теорією валентності зв’язків (англ. - bond valence model).

Принцип визначення формального заряду

Для того, щоб визначити формальний заряд на атомах металу MOSAEC аналізує його оточення. Для цього атом металу вилучається зі структури МОК, визначаються формальні заряди утворених фрагментів та їх сума пропорційно розподіляється між металами каркасу. Процес візуалізовано на Рис. 14.

Рис.14. Схема визначення формального заряду металу

Для визначення зарядів системи на першому етапі формальний заряд присвоюється всім неметалічним атомам. Ці заряди групуються в системи розподілених зарядів відповідно до принципів кон’югації та резонансу. Далі ці заряди розподіляються між металами структури відповідно до найбільш ймовірних ступенів окиснення. У випадку коли дана процедура не приводить

до задовільного результату, програма відмічає структуру певним класом помилки. Процес проілюстровано на Рис.15.

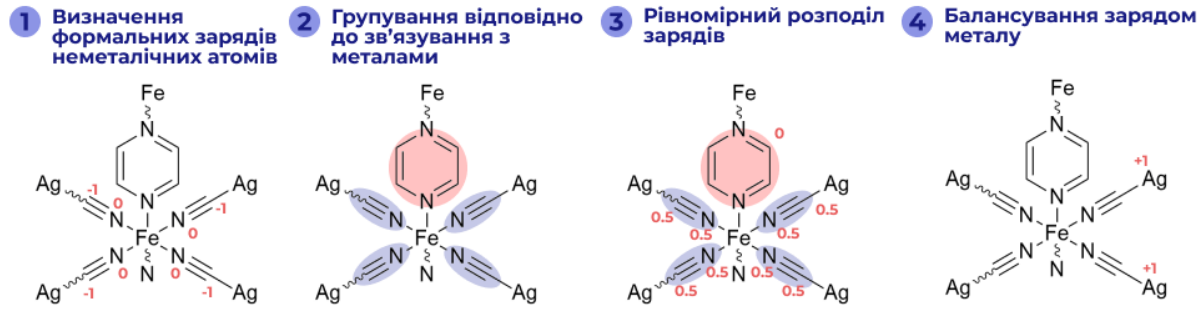


Рис.15. Візуалізація процесу розподілення зарядів в каркасі

В контексті нашої роботи нас цікавлять етапи 1-3 схеми, зображеної на Рис.15. Отримавши коректні формальні заряди на атомах ми можемо розрахувати заряд будь-якого фрагменту структури з досить великою точністю. Єдиною очевидною проблемою такого розрахунку будуть потенційні проблеми з розрахунком зарядів на метал-вмісних протийонах, адже заряд металу визначається з оточення, а перехідні метали можуть мати широкий спектр ступенів окиснення.

Розгляд використаних нами функцій програми MOSAEC

Для аналізу зарядів виділяється асиметрична одиниця каркасу, далі з неї ми отримуємо унікальні атоми за допомогою функції **get_unique_sites()** модуля **charge_assignment**. За допомогою функції **get_metal_sites()** ми отримуємо список металів структури, а функція **get_biding_sites()** надає нам список атомів, що напряму зв’язані з металами.

Для визначення формальних зарядів на атомах використовується емпіричний метод валентності зв’язків. Модель валентності зв’язків (англ. Bond valence model) – це теорія, що пояснює взаємозв’язок між валентністю йона, його координаційним числом, та довжиною зв’язку між йоном та його сусідами в кристалічній гратці. Ця модель широко застосовується в кристалографії для передбачення геометрії кристалічних структур. [42]

В роботі MOSAEC використовується спрощена ідеалізована версія цієї моделі, яка передбачає наступні відповідності: одинарний зв’язок –

валентність 1, подвійний – 2, потрійний – 3, зв’язок атом-метал має нульовий вклад (валентність 0). Ароматичні та делокалізовані зв’язки не мають цілочисельного значення валентності та розглядаються окремо залежно від системи. Відповідно до цього iVBS (англ. – idealized valence bond sum) будь-якого неметалічного атома є сумою таких умовних валентностей без урахування зв’язків з металами. Формальний заряд кожного з атомів далі отримується шляхом віднімання iVBS від кількості неспарених електронів структури Льюїса відповідного нейтрального атома. Якщо iVBS атома перевищує кількість неспарених електронів, використовується розширений шар електронів. Приклади зображено на Рис. 16.

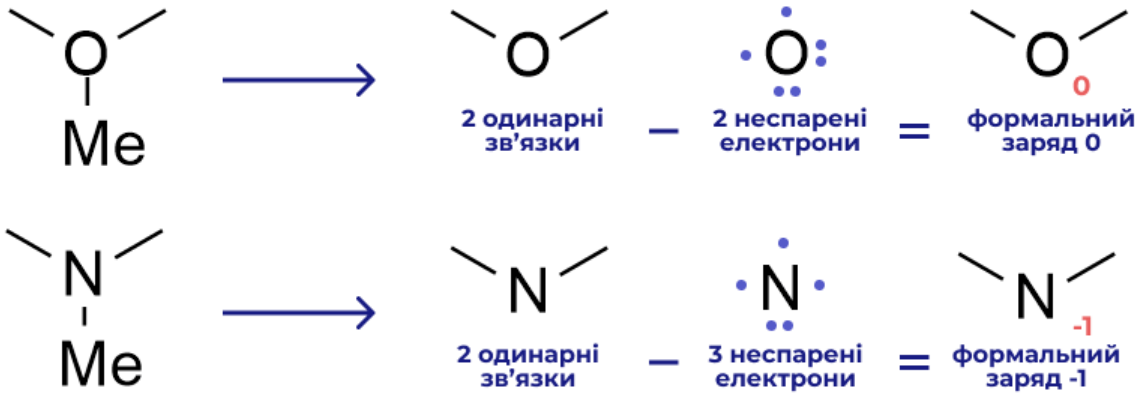


Рис.16. Приклади розрахунку формального заряду

Якщо розглянути виконання програми, спочатку вона аналізує структуру на наявність делокалізованих зв’язків за допомогою функції **delocalisedLBO()**. Вона повертає словник атомів, що входять до таких систем з VBS, що їм відповідає. Окремо аналізуються атоми циклічних систем за допомогою функції **ringVBOs()**. Ця функція забезпечує правильний аналіз ароматичних систем, та фрагментів, зв’язаних з ними, адже вони мають окреме розподілення зарядів, для того щоб максимально зберегти ароматичність.

Далі функція **iVBS_Oxidation_Contrib()** використовує попередні етапи для присвоєння формального заряду. Кількість валентних електронів визначається функцією **valence_e()** відповідно до властивостей кожного елемента. Деякі специфічні випадки, зокрема карбени, також передбачені і

визначаються окремо. Інформація повертається з функції у вигляді словника зі значенням формального заряду для кожного з атомів.

Далі нам необхідно надати відповідні формальні заряди усім атомам системи, адже вищезгаданий аналіз відбувається лише з асиметричною одиницею каркасу. Для цієї задачі використовується **redundantAON()**, що також повертає словник об’єктів типу ccdc.atom.Atom з відповідними значеннями.

2.5.6. Вилучення вільного розчинника

Вільний розчинник вилучається за допомогою функції **remove_free_solvent()** модуля **solvent_analysis.py**. У якості параметра вона отримує молекулярний об’єкт ccdc.molecule.Molecule та словник частковий заряд – ідентифікатор атома. Цей словник додатково формується функцією **get_rAON_atomlabels()**.

Компоненти молекулярного об’єкта перевіряються на наявність металів. Об’єкти, що містять метали та мають полімерні зв’язки вважаються частинами МОК, метал-вмісні протийони вилучаються з фіксацією заряду. Компоненти, що не містять металів та полімерних зв’язків також вилучаються з фіксацією заряду. Якщо заряд компоненту ненульовий, він записується до списку протийонів, якщо молекула нейтральна – до вільних розчинників.

Тут і надалі заряд компонентів рахується на основі часткових зарядів атомів за допомогою функції **get_component_charge()**.

```
def get_component_charge(rAON, component):
    """
    Takes out the charge of the individual atoms by searching for their
    labels in the rAON dictionary.
    Parameters:
        rAON (dict): dictionary atom label: oxidation contribution.
        component (ccdc.molecule.Molecule): disconnected component of the main molecule object.
Technicaly a ligand.
    Returns:
        charge (int or float): charge of the component.
    """
    charge = 0
    for atom in component.atoms:
        key = atom.label
        charge_unit = rAON.get(key)
        charge += charge_unit
    return charge
```

Відбувається це шляхом пошуку відповідних атомів в словнику rAON_atomlabels та отриманні суми зарядів для молекули.

Оскільки структури часто містять певні проблеми, для уникнення додаткових помилок ми додатково позначили, що ряд частинок мають мати нульовий заряд. До них відносяться: O (вода без воднів), N (аміак без воднів), H (частина «поламаної» води), O-O (невпорядкована вода без воднів), N-N (азот з неправильним порядком зв’язку), O-O-O (невпорядкована вода), D (частина «поламаної» дейтерованої води), OH (частіше за все вода). CO та CO_2 визначаються програмою MOSAEC як заряджені, тому вони також внесені до цього списку.

Під час виконання процесу вилучення розчинника також збирається статистика, яка буде поміщена в CSV файл. Вона збирається в словник statistics_output, який повертається функцією разом з очищеною молекулою, списками вільних розчинників та протийонів.

```
    statistics_output = {
        "free_solvents_output": free_solvents_output,
        "counterions_output": counterions_output,
        "charge_removed": charge_removed,
        "metal_counterion_flag": metal_counterion_flag,
        "huge_counterion_flag": huge_counterion_flag,
        "free_solvent_flag": free_solvent_flag,
        "counterions_flag": counterions_flag,
    }
```

Значення складових статистики будуть розглянуті в відповідному розділі. Якщо користувачем при запуску програми було вказано агрумент --

keep_bound, на цьому процес завершується та програма переходить до обчислення координат та створення нового CIF файлу.

2.5.7. Вилучення термінальних атомів оксигену

Вилучення термінальних атомів оксигену є нетривіальною задачею з якою не справляється жоден з існуючих алгоритмів. Під термінальним оксигеном мається на увазі атом оксигену, який зв'язаний лише з металом і є координованою молекулою води, якій не вистачає воднів. Приклад такого випадку зображено на Рис. 17.

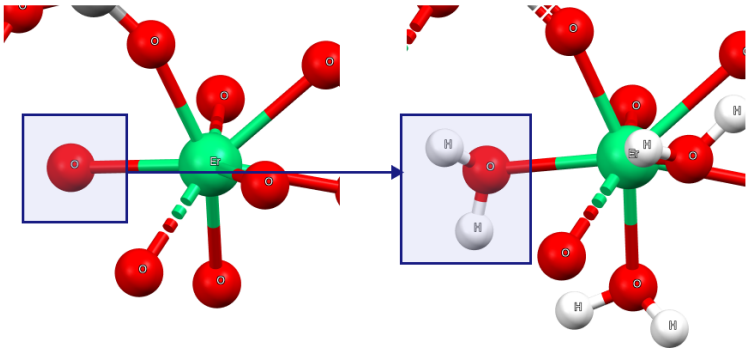


Рис.17. Приклад термінального атому окигену, який є водою

Теоретично це питання можна було б вирішити шляхом автоматичного додавання атомів гідрогену, проте термінальні атоми оксигену також можуть бути групами типу Me=O та Me-OH, що зустрічається досить часто.

Їх ідентифікація та вилучення відбувається за допомогою функції **get_oxo()** модуля **solvent_analysis**. Для того, щоб дізнатися чи є в структурі МОК оксо- атоми (Me=O, потрібні в структурі) програма звертається до CSD Entry - запису цього каркасу в базі CSD.

Для цього з назви файлу за допомогою функції **get_refcode()** визначається рефкод – код унікального запису структури в базі.

```
def get_refcode(file):
    """Gets the CSD entry refcode by searching the CCDC database. Works with filenames type
AAAAAA.cif
    and AAAAA_xxx.cif
    Parameters:
        file (str): filename *.cif
    Returns:
        ref_code (str): refcode identifier of the mof in CCDC
    """

    ref_code = file.replace(".cif", "")
    if "\\" or "/" in ref_code:
        ref_code = os.path.basename(ref_code)
    if "_" in ref_code:
        pos = ref_code.find("_")
        ref_code = ref_code[:pos]
        return ref_code
    return ref_code
```

Якщо структура в базі не присутня (або назва файлу не відповідає шаблону), всі ідентифіковані термінальні атоми оксигену будуть вилучені. Якщо користувач знає, що в його структурі таких немає, то він може залишити їх передавши при запуску програми аргумент --keep_oxo.

Функція **check_entry()** використовує отриманий рефкод для того, щоб отримати назву МОК з CSD Entry. Відбувається це шляхом текстового пошуку певних характерних фрагментів назви. Таким чином, якщо програма знайде згадування про атоми, які треба залишити, ця функція передасть відповідний terminal_oxo_flag.

Отже, отримавши цю інформацію, функція **get_oxo()** аналізує структуру на предмет наявності атомів кисню, що напряму зв’язані з металами. Якщо terminal_oxo_flag == True (наявні атоми Me=O), список металів, зв’язаних з киснем перевіряється на наявність найбільш вірогідних атомів (W, U, Mo, V, Np, Ti, список складено з особистих спостережень).

Якщо найбільш вірогідні метали присутні, атоми оксигену на них залишаються, а всі інші видаляються. Так ми можемо впоратися з ситуаціями, коли в молекулі присутні і термінальні оксигени-вода, і групи Me=O. Якщо найбільш вірогідних металів нема, всі оксигени залишаються.

У випадку коли в CSD Entry нема згадки про групи Me=O, всі кисні видаляються. Також вони видаляються у випадку неправильного/неіснуючого рефкоду, проте у цьому випадку користувач

попереджається про це через командний рядок та робиться відповідний запис в файлі статистики. Функція також повертає словник statistics_output з частиною статистики.

```
'WARNING: refcode is not found in CCDC - all oxygen atoms suspected of being water will be removed
if --keep_oxo was not passed as argument. This can be caused by inconsistend filename (refcode is
extracted from filename) or absence of your MOF in CCDC. Rename your file to AAAAAA.cif or
AAAAAA_xxx.cif and try again.
```

2.5.8. Вилучення зв’язаних розчинників

Для вилучення зв’язаних розчинників використовуються функції **define_solvents()** та **check_solvents()** модуля **solvent_analysis**.

Для початку роботи зі структури полімерного каркасу вилучаються метали за допомогою функції **remove_metals()**. В результаті утворюється велика кількість окремих компонентів. Вільним розчинником в першому наближенні є нейтральні фрагменти цієї системи. Список саме таких фрагментів повертає функція **define_solvents()**. Єдиним виключенням є компонент OH, що сам по собі за хімічною природою є зарядженим, проте в більшості випадків є водою, тому вноситься до списку нейтральних молекул.

Список нейтральних молекул далі перевіряється функцією **check_solvents()**. Вона ідентифікує нейтральні місткові молекули (зв’язані з двома металами), а також місткові атоми оксигену, які є важливими частинами структури та мають залишитися. Також функція залишає в структурі карбоніли, які ідентифікуються як нейтральні. Ці два пункти були повністю проігноровані попередніми алгоритмами.

2.5.9 Генерація координат атомів та експорт нового CIF файлу

Фінальний список розчинників збирається зі списків solvent_mols_checked, free_solvents, counterions, oxo_mols за допомогою функції **get_solvents_to_remove()**. Якщо цей список не пустий, це означає що розчинник було ідентифіковано та solvent_present_flag задається як True. В

цьому випадку програма має експортувати новий CIF файл, який не містить розчинників. Якщо розчинників немає, всі наступні кроки, описані в цьому розділі, пропускаються.

Спочатку для видалення розчинників та запису нового CIF файлу був використаний наступний підхід:

- Атоми розчинників вилучалися з молекулярного об’єкту МОК за допомогою перебору його атомів та порівняння їх ідентифікаторів зі списком ідентифікаторів, що відповідають атомам розчинників
- Цей молекулярний об’єкт експортувався у новий CIF файл за допомогою функціоналу CSD Python API –

```
with io.CrystalWriter('filename') as cif_writer:
  cif_writer.write(mol_name)
```

На жаль, в результаті тестування структур, отриманих таким чином, ми побачили, що в них часто присутні дублікати атомів, які роблять структуру непридатною для високопродуктивного скринінгу та не можуть бути усунені застосуванням якогось простого скрипта.

Саме тому нами було створено власний парсер CIF файлів, роботу якого буде описано в цьому розділі. Функції парсера відносяться до модуля **cif_manipulation**.

Першим етапом є генерація fractional coordinates кожного з атомів розчинника за допомогою функції **get_coordinates()**. Координати приводяться до універсального формату та 5 знаків після коми. Стандартизація є важливим етапом, тому що в подальшому координати порівнюються між собою як текст, а не числа. Функція повертає двовимірний масив, по 3 координати на кожен атом.

Новий CIF файл створюється за допомогою функції **output_cif()** модуля **outputs**. Вона створює копію вихідного файлу у вказаній користувачем папці. Наповнення цього файлу зчитується як текст і передається функції **remove_solvents_from_file()**, яка аналізує рядки CIF файлу, приводить їх до стандартного формату та шляхом порівняння

координат знаходить рядки, які відповідають атомам розчинника. Ці рядки видаляються, їх кількість фіксується для подальшого аналізу. Кількість вилучених атомів та відредагований текстовий вміст повертаються функцією та записуються в відповідні змінні. Копія CIF файлу, створена раніше, оновлюється відповідно до нового текстового наповнення. Працюючи з вже правильно відформатованим текстовим файлом ми мінімізуємо кількість артефактів та помилок.

2.5.10. Збереження файлів зі статистикою та значення його складових

Під час аналізу програмою першого файлу, нею створюється CSV файл, в який далі записується статистика вилучення розчинника для кожної зі структур. За це відповідає функція **export_res()** модуля **outputs.**

Якщо було передано аргумент --keep_bound було вилучено тільки вільний розчинник, утворюється файл “Free_solvent_removal_results.csv”, при повному вилученні він буде називатися “Solvent_removal_results.csv” та містити більш розширене наповнення.

Записи в файл робляться поступово по мірі виконання програми для збереження оперативної пам’яті, а також для того щоб користувач не втратив статистику у випадку раптового збою з будь-якої причини.

Наповнення файлу “Free_solvent_removal_results.csv”

Наповнення кожної з колонок розписане в Таблиці 2, додаткове роз’яснення нетривіальних пунктів буде подано нижче.

Назва колонки	Значення
CIF	Назва вихідного файлу
Solvent	YES/NO в залежності від того чи було видалено розчинник
Free_solvent	Список атомів, згрупованих по молекулам вільного розчинника
Number_of_free_solvent_molecules	Кількість молекул вільного розчинника
Counterions	Список атомів, згрупованих по молекулам протийонів

Number_of_counterions	Кількість молекул протийонів
Charge_removed	Заряд, який було вилучено з системи (може бути позитивним і негативним, або 0 відповідно до суми зарядів частинок)
Total_atoms	Всього атомів, які було ідентифіковано як атоми розчинника
Atoms_removed	Кількість атомів, видалених із файлу парсером
Atoms_match_flag	TRUE якщо Total atoms і Atoms removed не співпадають
Metal_counterion_flag	TRUE якщо присутній протийон, який містить метал
Huge_counterion_flag	TRUE якщо присутній протийон з зарядом більше 10

Таблиця 2. Пояснення колонок статистики “Free_solvent_removal_results.csv”

Деякі з пунктів потребують додаткового пояснення, оскільки вони вирішують досить специфічні проблеми.

Atoms_match_flag

Оскільки нами було написано власний парсер і він працює з використанням координат, для екстраординарних випадків ми вирішили передбачити перевірку того, що всі атоми розчинника були видалені правильно. Під час тестування програми не було знайдено випадків неспівпадінь, проте така помилка може бути критичною для подальших розрахунків і ми беремо таку можливість до уваги.

Metal_counterion_flag

Програма MOSAEC іноді неправильно визначає заряди протийонів з атомами перехідних металів, адже вони можуть мати широкий спектр ступенів окиснення. Такі структури додатково позначаються для подальшої перевірки.

Huge_counterion_flag

В структурах МОК зустрічаються специфічні протийони великого розміру. Вони мають кулеподібну форму і містять в собі багато атомів металів. Заряд таких структур програма визначає неправильно, тому вони позначаються в цій колонці. Приклад таких протийонів зображено на Рис. 18.

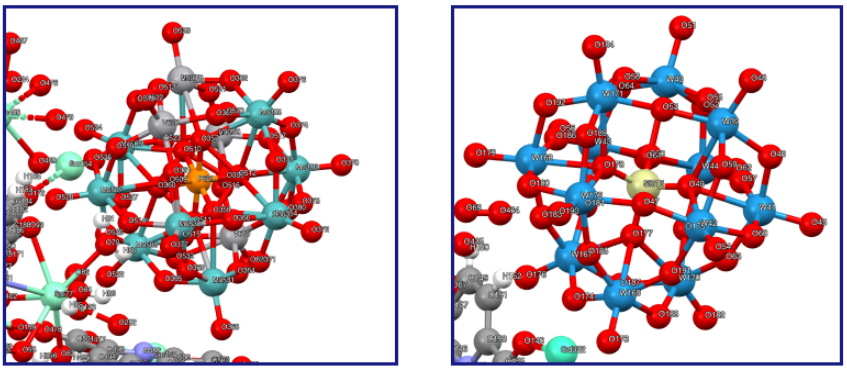


Рис. 18. Приклади кулеподібних протийонів

Наповнення файлу “Solvent_removal_results.csv”

Як вже було сказано раніше, цей файл експортується у випадку повного вилучення розчинника, тому він містить усі колонки, які присутні в Таблиці 2 і додаткову статистику, яка розписана у Таблиці 3.

Bound_solvent	Список атомів, згрупованих по молекулам зв’язаного розчинника
Number_of_bound_molecules	Кількість молекул зв’язаного розчинника
Flag_double	TRUE якщо біля атома, зв’язаного з металом є подвійні зв’язки
Flag_aromatic	TRUE якщо біля атома, зв’язаного з металом є ароматичні зв’язки
Terminal_oxo_flag	TRUE якщо було вилучено термінальні атоми оксигену
Entry_terminal_oxo	TRUE якщо в CSD Entry було знайдено згадку про групи Me=O, FAILED REFCODE якщо рефкод структури не було знайдено
OH_removed	Програма вилучає OH групи, зв’язані з металами, тому що статистично це найчастіше молекули води. Такі структури додатково позначаються TRUE щоб користувач про це знав.
Oxo_OH	TRUE якщо присутні термінальні оксигени на атомах U або Zr, ці атоми можуть бути OH групами, чи це так може визначити лише людина.

Таблиця 3. Пояснення колонок статистики “Solvent_removal_results.csv”

Наявність такої детальної статистики має велику цінність для користувачів, адже вона дозволяє оперативно перевіряти спірні моменти для забезпечення максимальної автоматизації процесу вилучення розчинника з мінімальним витраченим на це часом і людським вкладом.

2.5.11. Вивід командного рядка

Для зручності роботи з невеликими кількостями структур, а також контролю проходження процесу користувачем нами було передбачено вивід статистики в командний рядок. Туди ж виводяться попередження, які передбачені програмою, наприклад у випадку коли рефкод структури не був знайдений у базі CSD.

Вимкнути вивід командного рядка можна за допомогою передачі аргументів -v або --verbose при запуску програми.

РОЗДІЛ ІІІ

ВАЛІДАЦІЯ АЛГОРИТМУ ТА ВИЗНАЧЕННЯ ЙОГО ЕФЕКТИВНОСТІ В ПОРІВНЯННІ З ІСНУЮЧИМИ РІШЕННЯМИ

Для того, щоб довести, що наше рішення є ефективним, а також працює краще ніж ті, які існували до цього, необхідно провести комплексну валідацію нового алгоритму. На жаль, це можна зробити тільки шляхом візуального аналізу правильності вилучення розчинників.

3.1 Валідація алгоритму

Для валідації ми взяли 800 випадкових структур з бази даних CSD у симетрії P1. Вони були підготовані до вилучення розчинника шляхом виправлення невпорядкованих атомів. З усіх файлів було вилучено розчинник двома способами окремо: повне вилучення та вилучення лише вільного розчинника з аргументом --keep_bound. Шляхом візуальної оцінки

результатів та аналізу отриманих CSV файлів було зроблено висновки, загалом нами було проаналізовано 1600 структур. Далі в цьому розділі буде розглянуто статистичний аналіз отриманих результатів.

Аналіз результатів

Нами було проведено візуальний огляд 800 структур, з яких було вилучено весь розчинник. Таким чином ми економимо час, адже робити це вручну досить довго, а вилучення вільного розчинника ми можемо перевірити шляхом порівняння файлу статистики з файлом для вилучення розчинників всіх типів. Результати аналізу візуалізовано на Рис.19.

Рис.19. Візуалізація результатів аналізу даних

Бачимо, що розроблений алгоритм має високу ефективність та низький рівень помилок. У випадку вилучення вільного розчинника всі отримані структури були правильними (отриманий CIF файл не містив розчинника та не втратив ніяких зайвих атомів). При цьому правильний заряд був визначений у 98.8% випадків, лише 12 з 800 структур містили

помилки. При цьому всі структури, що містили помилки, були відмічені для повторної перевірки відповідно до типу помилки.

У випадку повного вилучення розчинників, 99.75% структур були правильними, відсоток МОК з проблемами заряду склав також 98.8%. Ці результати легко пояснити, адже зв'язаний розчинник представляє собою нейтральні молекули, весь вилучений зі структури заряд складають протийони.

Аналіз помилок алгоритму

У випадку зв'язаних розчинників причиною помилок стала досить нетривіальна ситуація – дві структури містили на атомах U і воду, і групи Me=O, що не ніяк не передбачено алгоритмом і в принципі на жаль не визначається ніяк окрім звернення до публікації, де було представлено цю структуру.

Вільні ж розчинники видаляються з 100% успіхом, проте є проблеми з зарядами, причому лише на протийонах, які містять метали. Це пояснюється тим, що MOSAEC визначає заряд металів за рахунок їх оточення, тому не завжди може правильно визначити заряд йона, адже метали можуть мати досить широкий спектр ступенів окиснення. Наприклад, для йона $MnBr_4$ (2-) заряд визначається як -4, що передбачає ступінь окиснення мангану 0, коли він має бути +2. Аналогічно з SbF_6 (-), заряд якого програмою визначається як -6. Цю проблему можна вирішити шляхом складання списку проблемних метал-вмісних протийонів для того, щоб в подальшому ці недоліки були виправлені.

Йому також не під силу правильне визначення зарядів у вищезгаданих кулеподібних протийонах, проте їх кількість досить незначна.

Варто відмітити, що всі структури з неправильно визначеним зарядом було додатково помічено за допомогою Metal_counterion_flag та Huge_counterion_flag. Перший позначає потенційно проблемні протийони з

металами, а другий показує МОК з кулеподібними протийонами. Це означає що всі ці помилки були б виправлені користувачем за мінімальний час.

Аналіз типів вилучених розчинників

Провівши валідацію алгоритму, ми зацікавились питанням аналізу розчинників та протийонів, які найчастіше зустрічаються у структурах валідаційного сету. Це було виконано шляхом аналізу таблиць статистики, отриманих в результаті виконання програми.

Вільні розчинники

Всього з 800 структур було вилучено 6152 молекул вільних розчинників, причому найбільш популярним був атом кисню O. який є водою без атомів гідрогену. Статистика найбільш популярних розчинників та їх кількості показано в Таблиці 4.

Список атомів	Кількість	Молекула
['O']	3054	O
['H', 'H', 'O']	1675	H_2O
['C', 'C', 'C', 'H', 'H', 'H', 'H', 'H', 'H', 'H', 'N', 'O']	300	DMF
['O', 'O']	254	O_2
['C', 'O', 'O']	93	CO_2
['N', 'N']	60	N_2
['C', 'H', 'H', 'H', 'H', 'O']	52	CH3OH
['C', 'C', 'C', 'C', 'H', 'H', 'H', 'H', 'H', 'H', 'H', 'H', 'H', 'N', 'O']	48	MeEtNOH
['C', 'C', 'H', 'H', 'H', 'H', 'H', 'H', 'O']	46	EtOH

Таблиця 4. Найбільш популярні вільні розчинники

Протийони

Всього алгоритмом було вилучено 1060 протийонів. Статистика кількості протийонів показана в Таблиці 5.

Список атомів	Кількість	Молекула
['C', 'C', 'H', 'H', 'H', 'H', 'H', 'H', 'H', 'H', 'N']	120	$CH_3CH_2NH_3$+
['Cl']	116	Cl-
['Cl', 'O', 'O', 'O', 'O']	102	ClO_4-
['N', 'O', 'O', 'O']	98	NO_3-

['C', 'C', 'C', 'C', 'H', 'H', 'H', 'H', 'H', 'H', 'H', 'H', 'H', 'H', 'H', 'N']	52	Me_3CHNH_2+
['B', 'F', 'F', 'F', 'F']	37	BF_4-
['F', 'F', 'F', 'F', 'F', 'F', 'P']	24	PF_6-
['Br']	20	Br-

Таблиця 5. Найбільш популярні види протийонів

Зв’язані розчинники

Загалом кількість вилученого зв’язаного розчинника склала 2579 молекул, в це число не входить кількість термінальних оксигенів, яка склала 1997 атомів. Статистика молекул зв’язаного розчинника показана в Таблиці 6

.

Список атомів	Кількість	Молекула
['H', 'H', 'O']	1748	H_2O
['C', 'C', 'C', 'H', 'H', 'H', 'H', 'H', 'H', 'H', 'N', 'O']	396	DMF
['C', 'C', 'C', 'C', 'H', 'H', 'H', 'H', 'H', 'H', 'H', 'H', 'H', 'N', 'O']	92	MeEtCNOH
['C', 'C', 'H', 'H', 'H', 'H', 'H', 'H', 'O', 'S']	54	DMSO
['H', 'O']	49	OH (water)
['C', 'H', 'H', 'H', 'H', 'O']	42	CH_3OH
['C', 'C', 'C', 'C', 'C', 'H', 'H', 'H', 'H', 'H', 'H', 'H', 'H', 'H', 'H', 'N', 'O']	36	Et_2CNOH
['C', 'C', 'H', 'H', 'H', 'H', 'H', 'H', 'O']	18	EtOH
['C', 'C', 'C', 'C', 'C', 'H', 'H', 'H', 'H', 'H', 'N']	16	піридин
['C', 'C', 'C', 'C', 'C', 'C', 'C', 'C', 'C', 'C', 'H', 'H', 'H', 'H', 'H', 'H', 'H', 'H', 'N', 'N']	16	біпіридин

Таблиця 6. Найбільш популярні види зв’язаних розчинників

Висновок з валідації програми

Виходячи з розгорнутого аналізу результатів ми можемо зробити висновок, що розроблений нами алгоритм є ефективним для стандартизації структур метал-органічних каркасів шляхом вилучення розчинника, тому що він має низький відсоток помилок та забезпечує майже повну автоматизацію процесу. Також нами передбачені спеціальні відмітки для того, щоб користувач мав можливість вручну перевірити потенційно помилкові

структури. Варто відмітити, що всі файли з неправильно визначеним зарядом були відмічені для додаткової перевірки.

3.2 Порівняння з існуючими алгоритмами

Для того, щоб оцінити переваги розробленого нами алгоритму над існуючими, нами було вирішено проаналізувати структури з помилково вилученим розчинником з баз даних CSD MOF Subset та CoRE MOF 2019 ASR (тут було обрано саме All Solvent Removed для можливості аналізу вилучення зв’язаних розчинників).

Для цього нами було обрано по 300 структур, які містили помилки пов’язані з вилученням розчинника. Відбір відбувався вручну, проте випадковим чином, шляхом аналізу баз за допомогою програми MOSAEC та візуального порівняння оригінальних структур з очищеними для підтвердження наявності помилки.

Отримавши список відповідних рефкодів ми викачали відповідні їм CIF файли з бази CSD у симетрії P1 за допомогою спеціального скрипта. Ці файли далі були оброблені за допомогою нашої програми, жоден з них не викликав помилок виконання коду. Далі шляхом ручної перевірки ми проаналізували правильність отриманих структур, результати аналізу отриманих даних будуть розглянуті нижче. Відсотки, подані у таблицях цього розділу можуть не давати у сумі 100%, адже деякі структури містять у собі декілька помилок одночасно.

Аналіз структур з CSD MOF Subset

На жаль у нас немає вичерпної статистики щодо кількості помилкових структур в цій базі даних, проте беручи до уваги співвідношення «хороших» до «поганих» МОК в проаналізованій нами за допомогою програми MOSAEC вибірці, ми можемо зробити припущення, що цей відсоток дуже високий.

У структурах, розглянутих нами, було виявлено ряд типів помилок, статистика яких поміщена в Таблиці 7.

Тип помилки	Кількість структур	% від 300
Вилучення заряджених лігандів	272	90,7%
Вилучення оксигенів з груп Me=O	26	8,7%
Інші проблеми	4	1,3%

Таблиця 7. Типи помилок в розглянутих 300 структурах CSD MOF Subset

Бачимо, що високий відсоток становить вилучення заряджених лігандів, що не дивно, адже їх алгоритмом передбачається видалення всіх монодентантних лігандів незалежно від заряду. Аналогічно часто відбувається з бідентантними лігандами, так званими «містковими», які в більшості випадків мають залишатися на місці. Приклад неправильного видалення можна побачити на Рис. 20. Me=O також видаляються без якогось додаткового аналізу, тому часто це є помилкою. На жаль, ці помилки, а особливо їх кількість, робить цю базу практично непридатною для високоточних розрахунків.

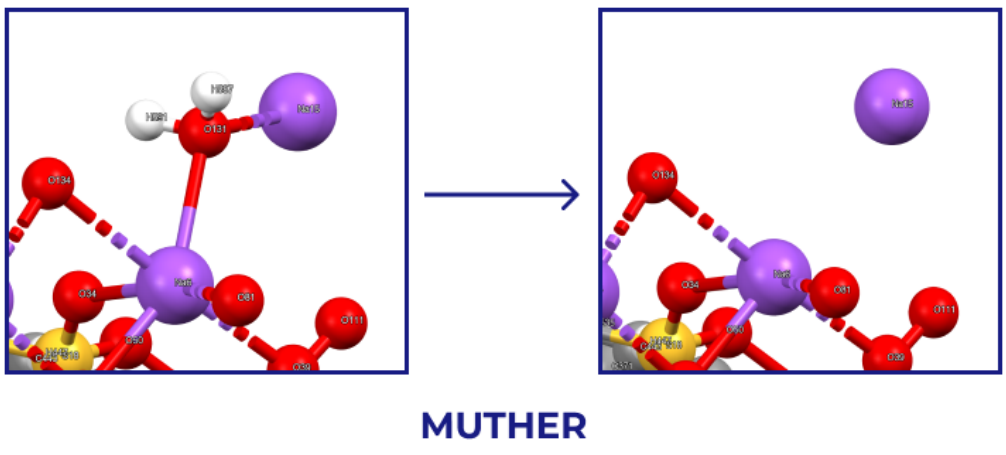


Рис. 202. Помилкове вилучення «місткової» води

З 300 структур кількість правильно оброблених каркасів за допомогою розробленого нами алгоритму склала 275, що відповідає 91,7%. З 25

помилкових структур (8,3%) ми змогли виділити основні типи помилок, показані в Таблиці 8.

Причина помилки	Кількість структур	% від всіх помилок (від загальної кількості)
Неправильний заряд через невпорядковані атоми	18	72% (6%)
Видалення комплексу металу	1	4% (0,3%)
Видалення потрібного нейтрального ліганду	5	20% (1,7%)
«Літаючі» атоми металу	1	4% (0,3%)

Таблиця 8. Типи помилок нового алгоритму

Більшість помилкових структур були отримані через значні недоліки вихідних CIF файлів, зокрема невпорядковані атоми.

Аналіз стукрур CoRE MOF 2019 ASR

Аналогічно до аналізу CSD MOF Subset ми проаналізували типи помилок в використаних нами структурах. Детальна статистика показана в Таблиці 9.

Тип помилки	Кількість структур	% від 300
Вилучення зайвих атомів	13	4,3%
Некоректне вилучення протийонів	35	11,7%
Вилучення кисню груп Me=O	69	23%
Вилучення заряджених лігандів	177	59%
Неправильне вилучення зв’язаного розчинника	2	0,7%
Вилучення металів з каркасу	11	3,7%
Неправильне вилучення вільного розчинника	8	2,7%

Таблиця 9. Типи помилок в 300 структурах CoRE MOF 2019 ASR

Шляхом аналізу структур, стандартизованих новим алгоритмом, визначено, що лише 53 з них мають помилки, що відповідає 17,7% від загальної кількості.

Типи помилок детальніше розглянуто у Таблиці 10.

Причина помилки	Кількість структур	% від всіх помилок (від загальної кількості)
Неправильний заряд/вилучення через невпорядковані атоми	46	86,8% (15,3%)
Неправильний заряд/вилучення через відсутність атомів водню	13	24,5% (4,3%)
Неправильне вилучення комплексів металів	2	3,8% (0,7%)

Таблиця 10. Типи помилок нового алгоритму

Висновки з порівняння з існуючими алгоритмами

Провівши аналіз вилучення розчинника з структур з баз CSD MOF Subset та CoRE MOF 2019 ASR, що містили помилки, бачимо, що запропонований нами алгоритм є більш ефективним. Головною його перевагою є аналіз зарядів, що дозволяє елімінувати великий відсоток помилкового вилучення заряджених лігандів, а також неправильний заряд каркасу через вилучення протийонів. Також важливим плюсом є аналіз термінальних атомів оксигену та бідентантних лігандів.

Значним обмеженням застосування нашого алгоритму на неідеальних експериментальних структурах є те, що невпорядковані атоми та пропущені атоми гідрогену можуть негативно впливати на вилучення окремих молекул або визначення заряду системи. Якщо проблема додавання атомів гідрогену може бути вирішена автоматично, то для приведення до коректного вигляду структур з невпорядкованими атомами на сьогоднішній день автоматизованих рішень немає.

ВИСНОВКИ

1. Нами було проведено детальний аналіз алгоритмів стандартизації структур метал-органічних каркасів в базах даних CSD MOF Subset та CoRE MOF 2019. Виділено основні недоліки, які мають бути втілені в розробці нового алгоритму для цієї задачі.
2. Аналіз структур в базі даних CoRE MOF 2019 показав, що близько 50% структур, що входять до неї, містять помилки, які роблять цю базу непридатною для проведення високоточних розрахунків. Значна частина цих помилок утворилась в результаті використання неефективного методу вилучення розчинника – 30.0 і 32.0 відсотки структур відповідно містять некоректний заряд, 8.4 та 3.4% містять помилкове видалення металів та лігандів.
3. Було розроблено новий алгоритм для вилучення розчинника та детально описано протокол його дії.
4. Валідація розробленого програмного забезпечення на 800 підготованих структурах показала ефективність 100% при вилученні вільного розчинника, 98,8% структур при цьому мали правильний заряд. При вилученні всього розчинника правильність структур становила 99,75%.
5. Додаткове порівняння з існуючими алгоритмами показало, що наш підхід є значно більш ефективним, зокрема завдяки аналізу зарядів фрагментів, термінальних оксигенів та бідентантних лігандів.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

[1] K. M. Jablonka, D. Ongari, S. M. Moosavi, and B. Smit, “Big-Data Science in Porous Materials: Materials Genomics and Machine Learning,” *Chem. Rev.*, vol. 120, no. 16, pp. 8066–8129, Aug. 2020, doi: 10.1021/acs.chemrev.0c00004.

[2] Y.-S. Bae and R. Q. Snurr, “Development and Evaluation of Porous Materials for Carbon Dioxide Separation and Capture,” *Angew. Chem. Int. Ed.*, vol. 50, no. 49, pp. 11586–11596, Dec. 2011, doi: 10.1002/anie.201101891.

[3] O. Kadioglu and S. Keskin, “Efficient separation of helium from methane using MOF membranes,” *Separation and Purification Technology*, vol. 191, pp. 192–199, Jan. 2018, doi: 10.1016/j.seppur.2017.09.031.

[4] T. Ghanbari, F. Abnisa, and W. M. A. Wan Daud, “A review on production of metal organic frameworks (MOF) for CO2 adsorption,” *Science of The Total Environment*, vol. 707, p. 135090, Mar. 2020, doi: 10.1016/j.scitotenv.2019.135090.

[5] R. Sahoo, S. Mondal, D. Mukherjee, and M. C. Das, “Metal–Organic Frameworks for CO_2 Separation from Flue and Biogas Mixtures,” *Adv Funct Materials*, vol. 32, no. 45, p. 2207197, Nov. 2022, doi: 10.1002/adfm.202207197.

[6] K. Meyer, M. Ranocchiari, and J. A. van Bokhoven, “Metal organic frameworks for photo-catalytic water splitting,” *Energy Environ. Sci.*, vol. 8, no. 7, pp. 1923–1937, 2015, doi: 10.1039/C5EE00161G.

[7] Z. Wu, Y. Li, C. Zhang, X. Huang, B. Peng, and G. Wang, “Recent advances in metal-organic-framework-based catalysts for thermocatalytic selective oxidation of organic substances,” *Chem Catalysis*, vol. 2, no. 5, pp. 1009–1045, May 2022, doi: 10.1016/j.checat.2022.02.010.

[8] J. Cao, X. Li, and H. Tian, “Metal-Organic Framework (MOF)-Based Drug Delivery,” *CMC*, vol. 27, no. 35, pp. 5949–5969, Oct. 2020, doi: 10.2174/0929867326666190618152518.

[9] P. Horcajada *et al.*, “Porous metal–organic-framework nanoscale carriers as a potential platform for drug delivery and imaging,” *Nature Mater*, vol. 9, no. 2,

pp. 172–178, Feb. 2010, doi: 10.1038/nmat2608.

[10] H. D. Lawson, S. P. Walton, and C. Chan, “Metal–Organic Frameworks for Drug Delivery: A Design Perspective,” *ACS Appl. Mater. Interfaces*, vol. 13, no. 6, pp. 7004–7020, Feb. 2021, doi: 10.1021/acsami.1c01089.

[11] S.-W. Lv, J.-M. Liu, C.-Y. Li, N. Zhao, Z.-H. Wang, and S. Wang, “A novel and universal metal-organic frameworks sensing platform for selective detection and efficient removal of heavy metal ions,” *Chemical Engineering Journal*, vol. 375, p. 122111, Nov. 2019, doi: 10.1016/j.cej.2019.122111.

[12] M. Jurcic *et al.*, “Sensing and Discrimination of Explosives at Variable Concentrations with a Large-Pore MOF as Part of a Luminescent Array,” *ACS Appl. Mater. Interfaces*, vol. 11, no. 12, pp. 11618–11626, Mar. 2019, doi: 10.1021/acsami.8b22385.

[13] X. Fang, B. Zong, and S. Mao, “Metal–Organic Framework-Based Sensors for Environmental Contaminant Sensing,” *Nano-Micro Lett.*, vol. 10, no. 4, p. 64, Oct. 2018, doi: 10.1007/s40820-018-0218-0.

[14] P. Kumar, A. Deep, and K.-H. Kim, “Metal organic frameworks for sensing applications,” *TrAC Trends in Analytical Chemistry*, vol. 73, pp. 39–53, Nov. 2015, doi: 10.1016/j.trac.2015.04.009.

[15] G. Wu, J. Huang, Y. Zang, J. He, and G. Xu, “Porous Field-Effect Transistors Based on a Semiconductive Metal–Organic Framework,” *J. Am. Chem. Soc.*, vol. 139, no. 4, pp. 1360–1363, Feb. 2017, doi: 10.1021/jacs.6b08511.

[16] Y. J. Colón and R. Q. Snurr, “High-throughput computational screening of metal–organic frameworks,” *Chem. Soc. Rev.*, vol. 43, no. 16, pp. 5735–5749, 2014, doi: 10.1039/C4CS00070F.

[17] Y. G. Chung *et al.*, “Computation-Ready, Experimental Metal–Organic Frameworks: A Tool To Enable High-Throughput Screening of Nanoporous Crystals,” *Chem. Mater.*, vol. 26, no. 21, pp. 6185–6192, Nov. 2014, doi: 10.1021/cm502594j.

[18] P. Z. Moghadam *et al.*, “Development of a Cambridge Structural Database Subset: A Collection of Metal–Organic Frameworks for Past, Present, and Future,”

Chem. Mater., vol. 29, no. 7, pp. 2618–2625, Apr. 2017, doi: 10.1021/acs.chemmater.7b00441.

[19] S. R. Hall, F. H. Allen, and I. D. Brown, "The crystallographic information file (CIF): a new standard archive file for crystallography," *Acta Crystallogr A Found Crystallogr*, vol. 47, no. 6, pp. 655–685, Nov. 1991, doi: 10.1107/S010876739101067X.

[20] C. F. Macrae *et al.*, "*Mercury* : visualization and analysis of crystal structures," *J Appl Crystallogr*, vol. 39, no. 3, pp. 453–457, Jun. 2006, doi: 10.1107/S002188980600731X.

[21] "Accelrys (2016) Materials Studio. http://accelrys.com/products/collaborative-science/biovia-materials-studio/."

[22] K. Momma and F. Izumi, "*VESTA 3* for three-dimensional visualization of crystal, volumetric and morphology data," *J Appl Crystallogr*, vol. 44, no. 6, pp. 1272–1276, Dec. 2011, doi: 10.1107/S0021889811038970.

[23] O. V. Dolomanov, L. J. Bourhis, R. J. Gildea, J. A. K. Howard, and H. Puschmann, "*OLEX2* : a complete structure solution, refinement and analysis program," *J Appl Crystallogr*, vol. 42, no. 2, pp. 339–341, Apr. 2009, doi: 10.1107/S0021889808042726.

[24] M. D. Hanwell, D. E. Curtis, D. C. Lonie, T. Vandermeersch, E. Zurek, and G. R. Hutchison, "Avogadro: an advanced semantic chemical editor, visualization, and analysis platform," *J Cheminform*, vol. 4, no. 1, p. 17, Dec. 2012, doi: 10.1186/1758-2946-4-17.

[25] F. H. Allen, "The Cambridge Structural Database: a quarter of a million crystal structures and rising," *Acta Crystallogr B Struct Sci*, vol. 58, no. 3, pp. 380–388, Jun. 2002, doi: 10.1107/S0108768102003890.

[26] "CCDC Home | CCDC." https://www.ccdc.cam.ac.uk/ (accessed Apr. 02, 2023).

[27] H. Daglar and S. Keskin, "Recent advances, opportunities, and challenges in high-throughput computational screening of MOFs for gas separations," *Coordination Chemistry Reviews*, vol. 422, p. 213470, Nov. 2020, doi:

10.1016/j.ccr.2020.213470.

[28] E. Haldoupis, S. Nair, and D. S. Sholl, “Efficient Calculation of Diffusion Limitations in Metal Organic Framework Materials: A Tool for Identifying Materials for Kinetic Separations,” *J. Am. Chem. Soc.*, vol. 132, no. 21, pp. 7528–7539, Jun. 2010, doi: 10.1021/ja1023699.

[29] T. Van Heest, S. L. Teich-McGoldrick, J. A. Greathouse, M. D. Allendorf, and D. S. Sholl, “Identification of Metal–Organic Framework Materials for Adsorption Separation of Rare Gases: Applicability of Ideal Adsorbed Solution Theory (IAST) and Effects of Inaccessible Framework Regions,” *J. Phys. Chem. C*, vol. 116, no. 24, pp. 13183–13195, Jun. 2012, doi: 10.1021/jp302808j.

[30] T. Watanabe and D. S. Sholl, “Accelerating Applications of Metal–Organic Frameworks for Gas Adsorption and Separation by Computational Screening of Materials,” *Langmuir*, vol. 28, no. 40, pp. 14114–14128, Oct. 2012, doi: 10.1021/la301915s.

[31] C. M. Simon *et al.*, “The materials genome in action: identifying the performance limits for methane storage,” *Energy Environ. Sci.*, vol. 8, no. 4, pp. 1190–1199, 2015, doi: 10.1039/C4EE03515A.

[32] Y. Basdogan, K. B. Sezginel, and S. Keskin, “Identifying Highly Selective Metal Organic Frameworks for CH_4/H_2 Separations Using Computational Tools,” *Ind. Eng. Chem. Res.*, vol. 54, no. 34, pp. 8479–8491, Sep. 2015, doi: 10.1021/acs.iecr.5b01901.

[33] S. Li, Y. G. Chung, and R. Q. Snurr, “High-Throughput Screening of Metal–Organic Frameworks for CO_2 Capture in the Presence of Water,” *Langmuir*, vol. 32, no. 40, pp. 10368–10376, Oct. 2016, doi: 10.1021/acs.langmuir.6b02803.

[34] C. M. Simon, R. Mercado, S. K. Schnell, B. Smit, and M. Haranczyk, “What Are the Best Materials To Separate a Xenon/Krypton Mixture?,” *Chem. Mater.*, vol. 27, no. 12, pp. 4459–4475, Jun. 2015, doi: 10.1021/acs.chemmater.5b01475.

[35] C. Altintas and S. Keskin, “Computational screening of MOFs for C 2 H 6 /C 2 H 4 and C 2 H 6 /CH 4 separations,” *Chemical Engineering Science*, vol. 139, pp. 49–60, Jan. 2016, doi: 10.1016/j.ces.2015.09.019.

[36] “Case: How many MOFs are there in the CSD? - The Cambridge Crystallographic Data Centre (CCDC).” https://www.ccdc.cam.ac.uk/support-and-resources/support/case/?caseid=9833bd2c-27f9-4ff7-8186-71a9b415f012 (accessed Apr. 04, 2023).

[37] “ConQuest | CCDC.” https://www.ccdc.cam.ac.uk/solutions/software/conquest/ (accessed Apr. 04, 2023).

[38] Y. G. Chung *et al.*, “Advances, Updates, and Analytics for the Computation-Ready, Experimental Metal–Organic Framework Database: CoRE MOF 2019,” *J. Chem. Eng. Data*, vol. 64, no. 12, pp. 5985–5998, Dec. 2019, doi: 10.1021/acs.jced.9b00835.

[39] L. E. Kreno, K. Leong, O. K. Farha, M. Allendorf, R. P. Van Duyne, and J. T. Hupp, “Metal–Organic Framework Materials as Chemical Sensors,” *Chem. Rev.*, vol. 112, no. 2, pp. 1105–1125, Feb. 2012, doi: 10.1021/cr200324t.

[40] S. R. Bahn and K. W. Jacobsen, “An object-oriented scripting interface to a legacy electronic structure code,” *Comput. Sci. Eng.*, vol. 4, no. 3, pp. 56–66, Jun. 2002, doi: 10.1109/5992.998641.

[41] “CSD Python API | CCDC.” https://www.ccdc.cam.ac.uk/solutions/software/csd-python/ (accessed Apr. 05, 2023).

[42] I. D. Brown, “Recent Developments in the Methods and Applications of the Bond Valence Model,” *Chem. Rev.*, vol. 109, no. 12, pp. 6858–6919, Dec. 2009, doi: 10.1021/cr900053k.