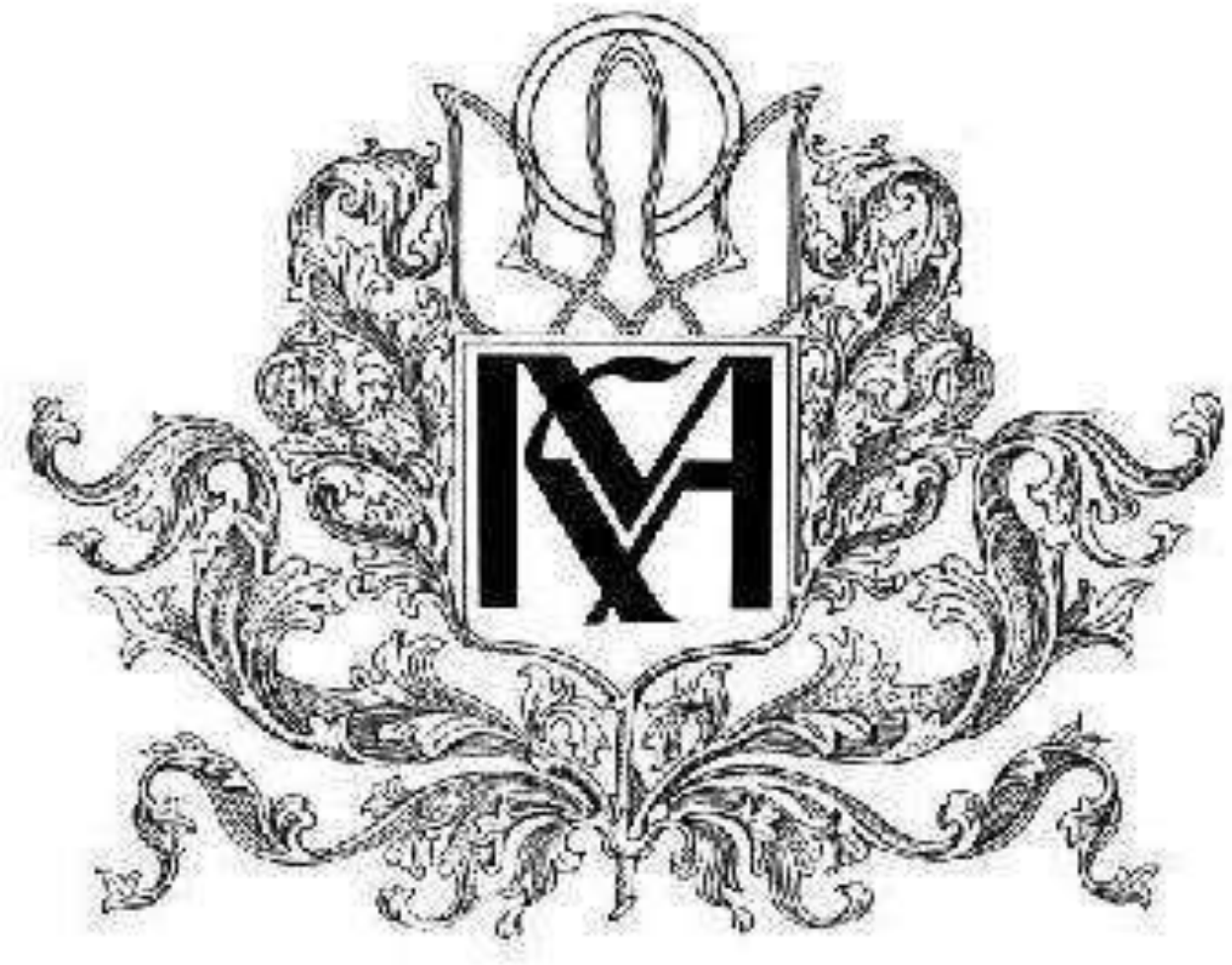
Використання фреймворку Lean для перевірки коректності математичних міркувань

Виконав студент 2 року навчання спеціальності 122 Комп’ютерні науки Кривошея О.О.

Керівник доцент, канд. ф.-м. наук Жежерун О.П.



Актуальність та Мета

- **Актуальність:**
  - Традиційні системи оцінюють лише кінцевий результат, ігноруючи логічний ланцюжок доведення.
  - Системи динамічної геометрії не здатні перевіряти дедуктивний математичний вивід.
  - Існує семантичний розрив між природномовною аргументацією учня та строгими системами комп’ютерної перевірки.
- **Мета дослідження:** Розробка архітектури програмного додатка, який використовує ядро Lean для автоматизованої перевірки геометричних міркувань учнів 7-9 класів, діагностує рівень їхньої підготовки та надає адаптивні рекомендації.

Об’єкт та предмет дослідження

1

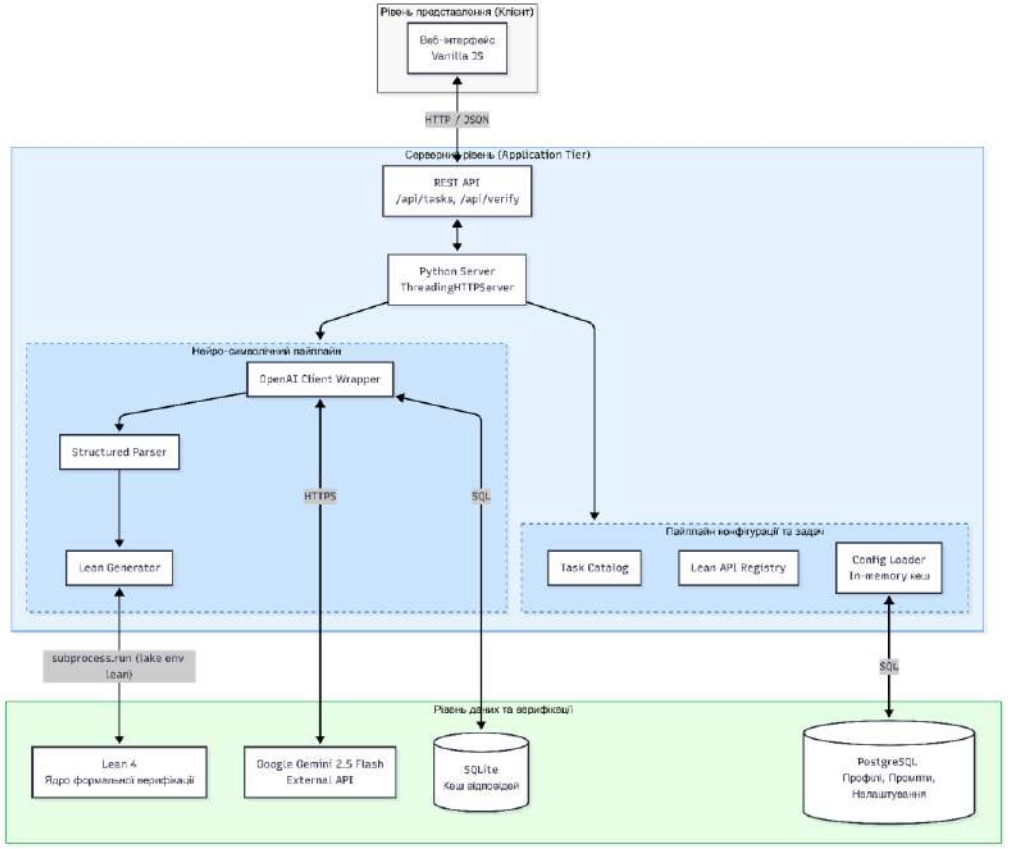
Об'єкт дослідження: процеси автоматизованої перевірки правильності математичних викладів у інтелектуальних навчальних середовищах.

2

Предмет дослідження: методи та програмні інструменти інтеграції середовища Lean у рекомендаційну систему.

Архітектура системи

- **Presentation Tier:** Тонкий клієнт (Vanilla JS) із механізмом FIFO-черги.
- **Application Tier:** Оркестратор на Python. Керує двома пайплайнами: обробки тексту та конфігурації задач.
- **Data & Verification Tier:** Ядро Lean 4, PostgreSQL (каталог задач), SQLite (кеш LLM).



Формалізація геометричного ядра в Lean 4

- **Прагматичний компроміс типів:** Логічні відношення перевіряються строго над полем раціональних чисел, евклідові метрики — через Float.
- **Безпека часткових операцій:** Використання монади Option, наприклад, для перетину паралельних прямих, гарантує перевірку винятків на рівні типів компілятора.
- **Модульність:** Ядро поділено на Core, Predicates, Constructors та Metrics.

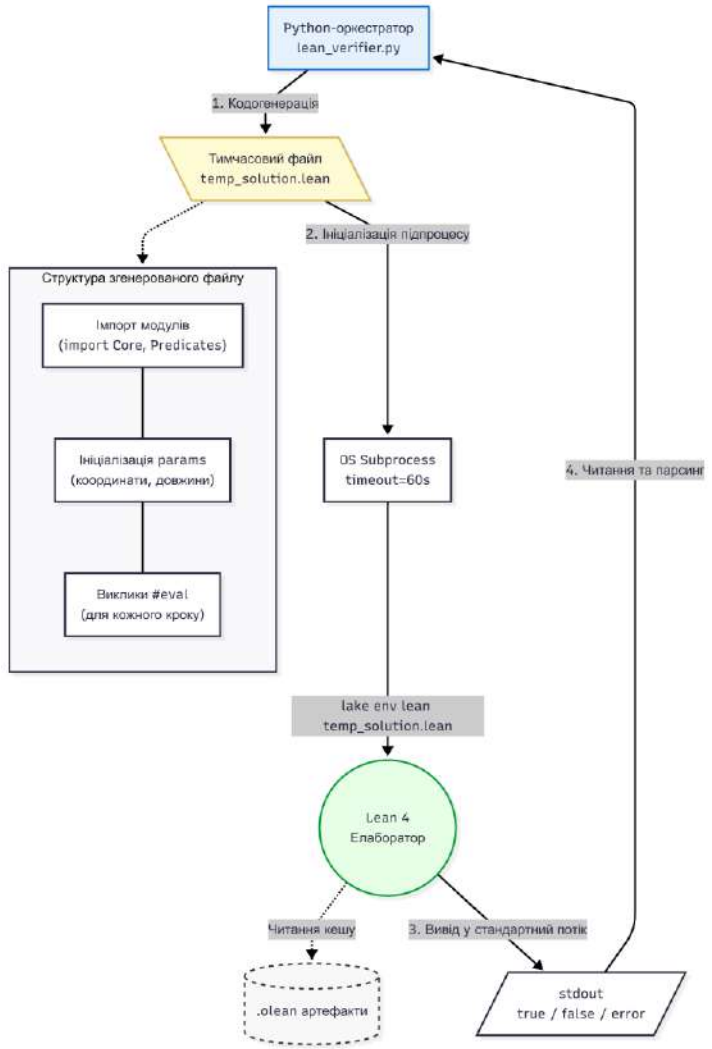
```
def intersection (l1 l2 : Line) : Option Point :=
  let det := l1.a * l2.b - l2.a * l1.b
  if det == 0 then
    none -- Прямі паралельні, унікальної точки перетину не існує
  else
    some { x :=..., y :=... }

structure Point where
  x : ℚ
  y : ℚ

structure Line where
  a : ℚ
  b : ℚ
  c : ℚ
Інваріант: a і b не можуть бути одночасно нулями
```

Пайплайн динамічної перевірки розв'язків

- **Нейро-символічна інтеграція:** LLM, наплад Gemini 2.5 Flash, вилучає факти з тексту учня та генерує формальний код.
- **Динамічна збірка:** Python формує тимчасовий .lean файл із командами #eval для кожного кроку міркувань.
- **Ізольований процес:** Формальна верифікація згенерованого коду через lake env lean із таймаутом ОС у 60 секунд.
- **Самокорекція:** При помилці компілятора stderr повертається в LLM для автоматичного виправлення (до 5 спроб).



Алгоритм маршрутизації помилок

Фінальний вердикт базується на агрегації двох сигналів:

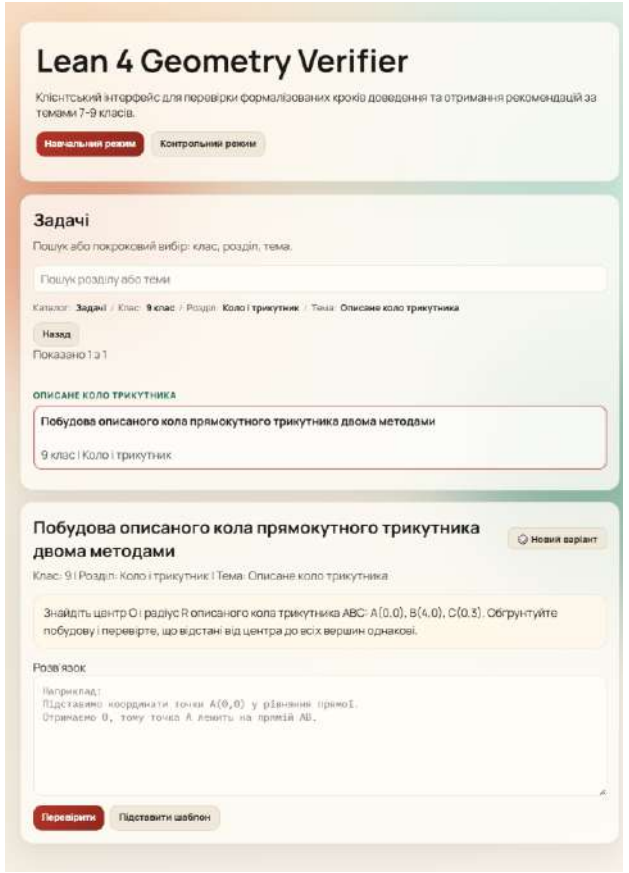
- **Математичний висновок Lean:** результати виконання команд #eval (pass/fail).
- **Семантичний NLP-аналіз:** порівняння введених кроків із матрицею requiredSteps з бази даних.

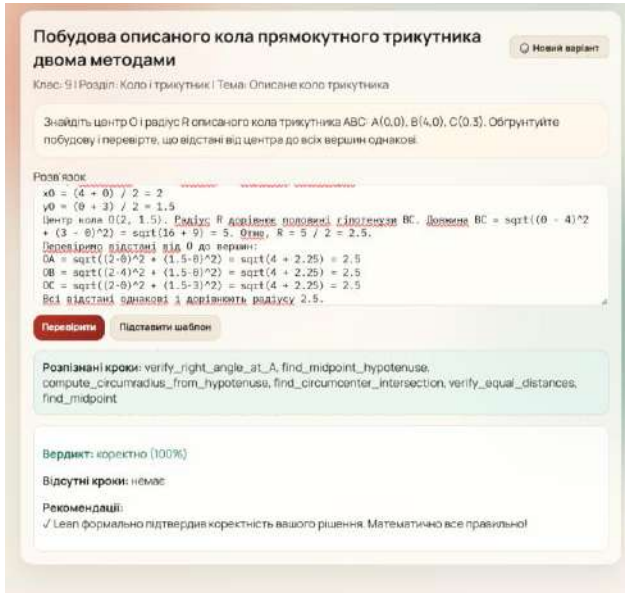
Сценарії оцінювання:

- **Correct (100%):** Всі математичні етапи пройдено, пропущених кроків 0.
- **Partial (10-90%):** Виявлено логічні помилки в міркуваннях, але базові кроки розпізнано.
- **Incorrect (0%):** Фатальна помилка critical_step_missing (наприклад, нерелевантний текст).

Користувацький інтерфейс: Успішне доведення

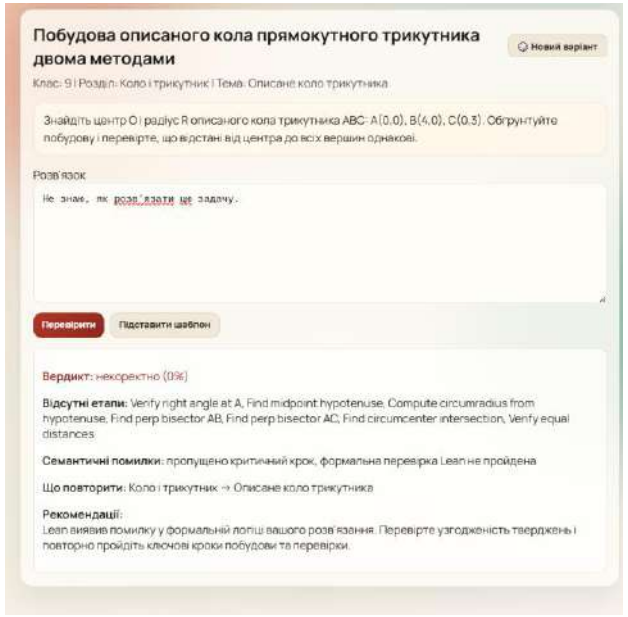
- Динамічна генерація параметрів задачі без вироджених конфігурацій.
- Миттєва візуалізація розпізнаних кроків .
- Повне підтвердження коректності ходу думок учня.

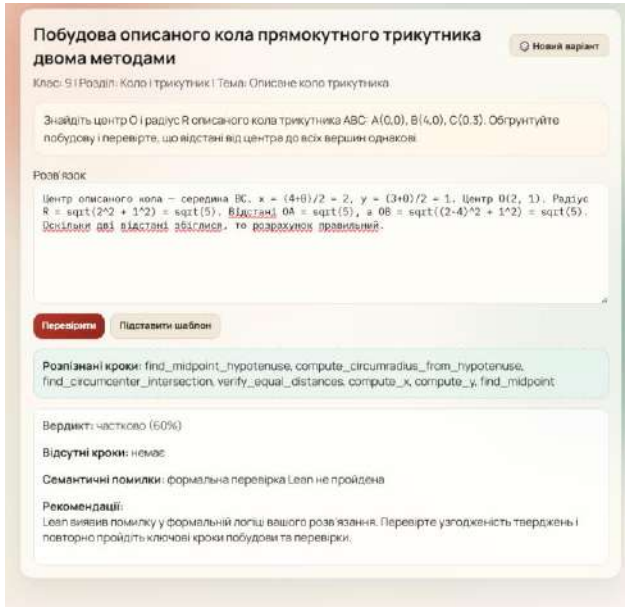




Користувацький інтерфейс: Формувальне оцінювання

- **Частковий бал:** Локалізація обчислювальної або логічної помилки в міркуваннях без повного відхилення розв'язку.
- **Критична помилка:** При беззмістовному введенні система формує масив "Відсутні етапи" та надає чітку педагогічну рекомендацію "Що повторити" з каталогу.





Оцінка ефективності та результати тестування

- **Відтворюваний процес контролю якості:**
  - **Smoke профіль:**
    - 198 успішних тестів.
    - Регресійні перевірки: 64 кейси.
    - Модульні тести: 14 кейсів.
    - Fuzz-тестування: 120 згенерованих перевірок.
  - **Масштабування:** Передбачено профілі Medium (500) та Heavy (3000) для стрес-тестування.
  - **Кешування:** SQLite-кеш для запитів до LLM знижує латентність під час повторних перевірок на 80%.

Висновки

- Доведено ефективність використання середовища Lean 4 для глибокої перевірки геометричних міркувань на противагу традиційним LMS.
- Розроблено геометричну бібліотеку з підтримкою строгих типів для гарантування детермінованості обчислень.
- Спроєктовано клієнт-серверну архітектуру нейро-символічної інтеграції, що вирішує проблему семантичного розриву.
- Впроваджено рушій маршрутизації помилок, який забезпечує якісне формувальне оцінювання учня.
- Працездатність прототипу підтверджена серією регресійних та fuzz-тестів. Система готова до використання як інтерактивний навчальний тренажер.

Дякую за увагу!

Можете поставити питання

Використання фреймворку Lean для перевірки коректності математичних міркувань

Виконав студент 2 року навчання спеціальності 122 Комп’ютерні науки Кривошея О.О.

Керівник доцент, канд. ф.-м. наук Жежерун О.П.