

**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**

**KHOA CÔNG NGHỆ THÔNG TIN**



**Sinh viên thực hiện: 19120557 - Trần Tuấn Kiệt**

**Giảng viên hướng dẫn: Lê Hoài Bắc**

**Nguyễn Ngọc Đức**

**LAB 02 - FIM**

|  |  |
| --- | --- |
| **CHẤM ĐIỂM** | |
| **Bằng số** | **Bằng chữ** |
|  |  |

**THÁNG 4 NĂM 2022**

**MỤC LỤC**

# Đánh giá mức độ hoàn thành các chức năng

|  |  |
| --- | --- |
| **Chức năng** | **Mức độ hoàn thành** |
| Đọc dữ liệu | 100% |
| Tìm các tập phổ biến | 100% |
| Tìm tập phổ biến đóng | 100% |
| Tìm tập phổ biến tối đại | 100% |
| Tìm luật kết hợp | 0% |
| Ý nghĩa | 0% |

# Ý nghĩa các file:

Trong thư mục source code có chứa:

* Thư mục data: 4 file data theo yêu cầu + 1 file “simple\_text.txt” để thực hiện việc debug
* File Frequent\_itemset.py: chứa class chính để lưu trữ và tạo các hàm để xử lý.
* File utility.py: chứa các hàm để phục vụ cho Frequen\_itemset.py
* File main.py: nơi gọi hàm

# Các bước thực hiện

## 1/ Ý tưởng chung:

Ta sẽ lấy fie simple\_text.txt để xem quá trình hoạt động. File này dựa trên silde lý thuyết.

Calendar

Description automatically generated

**Hình 1.1: Ảnh cắt từ slide**

A picture containing text, electronics, keyboard

Description automatically generated

**Hình 1.2: Chuyển về dạng số vì compare số sẽ nhanh hơn compare chuỗi**

## 2/ Đọc dữ liệu



|  |  |  |
| --- | --- | --- |
| **Input** | **Output** | **Ý nghĩa** |
| file\_name: đường dẫn đến file  minSup: minimum support - số transaction tối thiểu. Nếu điền trường này thì không điền minSup\_percentage và ngược lại  minSup\_percentage: phần trăm số transaction tối thiểu (0 <= … <= 1). | không | Đọc file và lưu trữ vào biến self.dataset  Chuyển sang self.tidset. Vì ta sẽ duyệt nhiều lần trên self.dataset, cần chuyển sang tidset với độ phức tạp O(1) |

**Đọc dữ liệu**: khi điền minSup\_percentage = 0.5, chương trình sẽ tính minSup = Tổng số transaction \* minSup\_percentage (6\*0.5=3). Bất cứ itemset nào thỏa mãn minSup >= 3 đều sẽ được cho vào tập phổ biến self.freq\_itemset



**Hình 2.1: main.py – khởi tạo biến ds mang class Frequent\_itemset**

**Xem self.tidset:** gọi hàm “print(ds.tidset)”, chú ý phần tử index = 0 không có. Phần tử Index = 1 có mảng [1,3,4,5] tức là item 1 có trong transaction 1, 3, 4 và 5



**Hình 2.2: Lưu tidset dưới dạng list**

## 3/ Tìm tập phổ biến



**Lưu ý:** trong hàm trên có parameter debug chỉ dùng để theo dõi cách hoạt động, còn khi ta chỉ mong muốn kết quả thì không sử dụng biến này.

**Mã giả:**

Hàm get\_freq\_itemset:

freq\_set1 = Tạo\_freq\_set\_gồm\_1\_item(tidset)

nếu freq\_set1 rỗng:

dừng, trả kết quả

freq\_set2 = Tạo\_freq\_set\_gồm\_1\_item(freq\_set1)

nếu freq\_set2 rỗng:

dừng, trả kết quả

while True:

freq\_set\_n+1 = Tạo\_freq\_set\_gồm\_n\_item(freq\_set\_n)

nếu freq\_set\_n+1 rỗng:

dừng, trả kết quả

Kết quả trả về là giao của các tập: freq\_set1, freq\_set2, freq\_set\_n, …

Để dễ theo dõi, ta sẽ dùng hàm self.debug() thay thế cho hàm trên: Text

Description automatically generated

**Hình 3.1: Kết quả của từng freq\_set\_n**

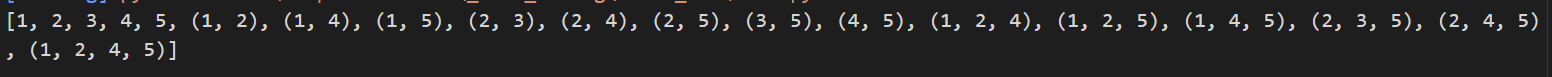
**Ví dụ:** freq\_set4 có cấu trúc [[(1, 2, 4, 5), [1, 3, 5]]] nghĩa là tập phổ biến gồm 4 items, xuất hiện ở transaction 1, 3, 5. Kiểm tra thủ công bằng tay, ta thấy kết quả ở trên đúng.

**Lưu ý:** biến debug = True chỉ dùng cho 1 mục đích ở trên (vì có thể gây lỗi do cấu trúc không nhất quán). Nếu ta sử dụng thêm các hàm khác, cần chạy lại với debug = False.

Text

Description automatically generated

**Hình 3.2: chạy lại nhưng với debug = False (không sử dụng debug)**



**Hình 3.3: Các tập phổ biến. Giống kết quả với hình 3.1 nhưng khác cấu trúc mảng**

## 4/ Tập phổ biến đóng



Tập phổ biến đóng là tập phổ biến mà không có tập nào bao nó có cùng độ phổ biến.



**Hình 4.1: Kết quả**

## 5/ Tập phổ biến tối đại





**Hình 5.1: Kết quả**