ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

KHOA KỸ THUẬT MÁY TÍNH

HỌ VÀ TÊN SINH VIÊN

Lê Trần Mạnh Tâm

BÁO CÁO ĐỒ ÁN MÔN HỌC

PHẦN MỀM RÚT GỌN BẢNG TRẠNG THÁI TỰ ĐỘNG BẰNG PHƯƠNG PHÁP PHÂN HOẠCH

GIẢNG VIÊN HƯỚNG DẪN

Ths. Trương Văn Cương

THÀNH PHỐ HỒ CHÍ MINH, 2023

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM

ĐỘC LẬP – TỰ DO – HẠNH PHÚC

CHI TIẾT

TÊN ĐỒ ÁN TIẾNG VIỆT: PHẦN MỀM RÚT GỌN BẢNG TRẠNG THÁI	
TÊN ĐỒ ÁN TIẾNG ANH: STATE TABLE REDUCTION SOFTWARE	
Giảng viên hướng dẫn: Ths. Trương Văn Cương, Khoa Kỹ thuật máy tính	
Thời gian thực hiện: Từ ngày: 8/5/2023 Đến ngày: 22/5/2023	
Sinh viên thực hiện: Lê Trần Mạnh Tâm	
Ngôn ngữ lập trình: Python **Phần mềm hỗ trợ:** Excel	
Chứng nhận của giảng viên (chữ ký và họ tên đầy đủ)	**Tp. Hồ Chí Minh, 27/2/2023** **Sinh viên** (chữ ký và họ tên đầy đủ)

MỤC LỤC

CHƯƠNG 1: GIỚI THIỆU ĐỀ TÀI 1
1.1 Giới thiệu 1
CHƯƠNG 2: NGÔN NGỮ LẬP TRÌNH PYTHON 1
2.1 Giới thiệu ngôn ngữ Python 1
2.2 Giới thiệu thư viện Pandas và một số hàm được sử dụng 1
2.2.1 DataFrame 2
2.2.2 Lệnh read_excel() 2
2.2.3 Truy xuất cột 3
2.2.4 Truy xuất hàng 3
2.4.5 Hàm index() 4
2.4.6 Hàm replace() 5
2.4.6 Hàm drop_duplicates() 5
CHƯƠNG 3: THUẬT TOÁN VÀ CÁC CHƯƠNG TRÌNH 7
3.1 Lưu đồ thuật toán 7
3.2 Mô phỏng bước 1 gom nhóm theo ngõ ra 8
3.3 Mô phỏng bước 2 rút gọn 9
3.4 Mô phỏng bước 3 tạo nhóm mới 10
3.5 Mô phỏng bước 4 lặp lại cho đến khi không còn rút gọn được nữa 11
3.6 Chương trình 12
CHƯƠNG 4: THỰC NGHIỆM 13
4.1 Môi trường thực nghiệm 13
4.2 Kết quả thu được 13
CHƯƠNG 5: ĐÁNH GIÁ 18
5.1 Độ chính xác 18
5.2 Thời gian đáp ứng 18
5.3 Một số hạn chế còn tồn tại 18
5.4 Kết luận 19
TÀI LIỆU THAM KHẢO 19

DANH MỤC HÌNH ẢNH

Hình 2. 1 Hàm read_excel() 2
Hình 2. 2 Hàm read_excel() 3
Hình 2. 3 Hàm iloc[] truy xuất cột 3
Hình 2. 4 Hàm loc[] 3
Hình 2. 5 Hàm iloc[] truy xuất hàng 4
Hình 2. 6 Hàm index() 4
Hình 2. 7 Hàm replace() 5
Hình 2. 8 Hàm replace() 5
Hình 2. 9 Hàm drop_duplicates() 6
Hình 2. 10 Hàm drop_duplicates() 6
Hình 3. 1 Lưu đồ thuật toán 7
Hình 3. 2 Bảng dữ liệu tham khảo 8
Hình 3. 3 Kết quả mô phỏng bước 1 9
Hình 3. 4 Bảng rút gọn thủ công 9
Hình 3. 5 Kết quả mô phỏng bước 2 10
Hình 3. 6 Kết quả mô phỏng bước 3 11
Hình 3. 7 Chương trình hoàn chỉnh 11
Hình 3. 8 Kết quả sau khi rút gọn 12

CHƯƠNG 1: GIỚI THIỆU ĐỀ TÀI

1.1 Giới thiệu

Đề tài "Rút gọn bảng trạng thái" là một chủ đề quan trọng trong lĩnh vực khoa học máy tính. Bảng trạng thái là một công cụ quan trọng để mô hình hóa hệ thống. Tuy nhiên, bảng trạng thái có thể trở nên rất lớn và phức tạp khi mô hình hóa các hệ thống lớn hơn. Do đó, rút gọn bảng trạng thái là một phương pháp để giảm thiểu số lượng trạng thái cần phải xử lý, từ đó giảm chi phí tính toán và tăng hiệu quả trong các ứng dụng thực tế. Trong đề tài này, em sẽ giới thiệu các phương pháp rút gọn bảng trạng thái bằng phương pháp phân hoạch.

CHƯƠNG 2: NGÔN NGỮ LẬP TRÌNH PYTHON

2.1 Giới thiệu ngôn ngữ Python

Python là một ngôn ngữ lập trình thông dịch, đa năng, dễ học và sử dụng. Python được phát triển bởi Guido van Rossum vào năm 1991 và được cung cấp miễn phí cho cộng đồng người dùng. Python có thể chạy trên nhiều nền tảng khác nhau bao gồm Windows, Linux, MacOS, và các hệ điều hành khác.

Python là ngôn ngữ lập trình được ưa chuộng trong nhiều lĩnh vực, bao gồm khoa học dữ liệu, trí tuệ nhân tạo, web development, game development và nhiều lĩnh vực khác. Python có cú pháp đơn giản, dễ đọc và dễ hiểu, làm cho việc phát triển phần mềm nhanh chóng và dễ dàng hơn. Ngoài ra, Python còn có một cộng đồng phát triển mạnh mẽ, cung cấp nhiều thư viện và framework hữu ích để giúp các lập trình viên phát triển ứng dụng một cách nhanh chóng và hiệu quả.

2.2 Giới thiệu thư viện Pandas và một số hàm được sử dụng

Thư viện Pandas là một trong những thư viện quan trọng và phổ biến nhất của Python dùng để xử lý dữ liệu. Pandas là viết tắt của "Panel Data" - một thuật ngữ trong thống kê, đại diện cho dữ liệu có cấu trúc theo dạng bảng.

Pandas cung cấp các đối tượng dữ liệu linh hoạt và dễ sử dụng như Series, DataFrame và Panel để thực hiện các phép xử lý dữ liệu như lọc, sắp xếp, thống kê, trực quan hóa dữ liệu, và nhiều hơn nữa. Ngoài ra, Pandas còn hỗ trợ đọc và ghi các định dạng dữ liệu phổ biến như CSV, Excel, SQL và JSON.

Một trong những ứng dụng phổ biến của Pandas là trong lĩnh vực khoa học dữ liệu, nơi nó được sử dụng để xử lý và phân tích dữ liệu. Pandas cũng được sử dụng rộng rãi trong các ứng dụng web để xử lý dữ liệu đầu vào và đầu ra.

Với tất cả các tính năng và đặc điểm của nó, Pandas là một thư viện rất mạnh mẽ và hữu ích cho các lập trình viên Python.

2.2.1 DataFrame

DataFrame là một trong những đối tượng dữ liệu quan trọng nhất trong thư viện Pandas của Python. DataFrame là một bảng dữ liệu hai chiều (hàng và cột) giống như trong Excel hoặc SQL, và nó được sử dụng để thực hiện các phép xử lý dữ liệu phức tạp.

Một DataFrame trong Pandas có thể được tạo ra từ nhiều nguồn dữ liệu khác nhau như Series, list, dict, và các file dữ liệu phổ biến như CSV, Excel, SQL và JSON.

DataFrame có thể chứa dữ liệu văn bản, số, datetime và các kiểu dữ liệu khác.

DataFrame cung cấp nhiều tính năng hữu ích để xử lý dữ liệu, bao gồm:

- Lọc dữ liệu dựa trên điều kiện.
- Sắp xếp dữ liệu theo một hoặc nhiều cột.
- Tính toán các thống kê mô tả như trung bình, độ lệch chuẩn, min, max, và nhiều hơn nữa.
- Trực quan hóa dữ liệu bằng các biểu đồ và biểu đồ phân tán.
- Gom nhóm dữ liệu theo một hoặc nhiều cột.

DataFrame trong Pandas là một trong những công cụ quan trọng và phổ biến nhất cho các lập trình viên và nhà khoa học dữ liệu để xử lý và phân tích dữ liệu.

2.2.2 Lệnh read_excel()

Lệnh read_excel() trong thư viện Pandas của Python được sử dụng để đọc dữ liệu từ một file Excel và chuyển đổi nó thành một đối tượng DataFrame.

Cú pháp của lệnh read_excel() như sau:

```
import pandas as pd

df = pd.read_excel('filename.xlsx')
```

Hình 2. 1 Hàm read_excel()

Trong đó, 'filename.xlsx' là tên của file Excel cần đọc. Nếu file Excel có nhiều sheet, bạn có thể chỉ định sheet cụ thể bằng cách sử dụng tham số sheet_name. Ví dụ:

```
df = pd.read_excel('filename.xlsx', sheet_name='Sheet1')
```

Hình 2. 2 Hàm read_excel()

Một số tham số khác của lệnh read_excel() bao gồm:
- header: chỉ định hàng nào trong tệp Excel là hàng tiêu đề.
- index_col: chỉ định cột nào trong tệp Excel là cột chỉ mục.
- usecols: chỉ định danh sách các cột cần đọc.
- skiprows: chỉ định số lượng hàng cần bỏ qua từ đầu tệp Excel.

Với lệnh read_excel(), bạn có thể dễ dàng đọc dữ liệu từ file Excel và sử dụng nó trong các phép tính và xử lý dữ liệu bằng thư viện Pandas của Python.

2.2.3 Truy xuất cột

Để truy xuất một cột trong Pandas Excel, bạn có thể sử dụng cú pháp sau:

```
import pandas as pd

# Đọc file Excel và chuyển đổi nó thành DataFrame
df = pd.read_excel('filename.xlsx')

# Truy xuất một cột bằng tên cột
col = df['column_name']

# Hoặc truy xuất một cột bằng số thứ tự của cột
col = df.iloc[:, column_index]
```

Hình 2. 3 Hàm iloc[] truy xuất cột

Trong đó, 'filename.xlsx' là tên của file Excel cần đọc, 'column_name' là tên của cột bạn muốn truy xuất và 'column_index' là số thứ tự của cột bạn muốn truy xuất. Lưu ý rằng khi truy xuất một cột bằng tên cột, kết quả trả về sẽ là một Series, còn khi truy xuất bằng số thứ tự của cột, kết quả trả về sẽ là một DataFrame mới chỉ chứa cột được truy xuất.

2.2.4 Truy xuất hàng

Để truy xuất một hàng trong Pandas Python, bạn có thể sử dụng một trong hai phương pháp sau:

1. Sử dụng phương thức loc[]: Phương thức này cho phép bạn truy xuất một hàng bằng tên chỉ mục của nó. Ví dụ:

```
import pandas as pd

# Đọc file Excel và chuyển đổi nó thành DataFrame
df = pd.read_excel('filename.xlsx')

# Truy xuất một hàng bằng tên chỉ mục của nó
row = df.loc['index_name']
```

Hình 2. 4 Hàm loc[]

Trong đó, 'filename.xlsx' là tên của file Excel cần đọc, 'index_name' là tên chỉ mục của hàng bạn muốn truy xuất.
2. Sử dụng phương thức iloc[]: Phương thức này cho phép bạn truy xuất một hàng bằng số thứ tự của nó. Ví dụ:

```
import pandas as pd

# Đọc file Excel và chuyển đổi nó thành DataFrame
df = pd.read_excel('filename.xlsx')

# Truy xuất một hàng bằng số thứ tự của nó
row = df.iloc[row_index]
```

Hình 2. 5 Hàm iloc[] truy xuất hàng

Trong đó, 'filename.xlsx' là tên của file Excel cần đọc, 'row_index' là số thứ tự của hàng bạn muốn truy xuất.
Lưu ý rằng kết quả trả về khi truy xuất một hàng sẽ là một Series, chứa các giá trị của các cột trong hàng đó.
Truy xuất một hàng trong Pandas Python là một phương pháp quan trọng để xử lý và phân tích dữ liệu bằng thư viện Pandas của Python.

2.4.5 Hàm index()

Hàm index() trong Pandas Python được sử dụng để trả về chỉ mục (index) của một giá trị hoặc một chuỗi giá trị trong một đối tượng Series hoặc DataFrame.
Cú pháp của hàm index() như sau:

```
import pandas as pd

# Tạo một Series
s = pd.Series([10, 20, 30, 40, 50], index=['a', 'b', 'c', 'd', 'e'])

# Trả về chỉ mục của giá trị 30 trong Series
idx = s.index[2]

# Tạo một DataFrame
data = {'name': ['John', 'Mary', 'Peter', 'Jane'],
        'age': [25, 30, 35, 40],
        'city': ['New York', 'London', 'Paris', 'Sydney']}
df = pd.DataFrame(data)

# Trả về chỉ mục của hàng có giá trị trong cột 'name' là 'Peter'
idx = df.index[df['name'] == 'Peter'][0]
```

Hình 2. 6 Hàm index()

Trong ví dụ trên, hàm index() được sử dụng để trả về chỉ mục của giá trị 30 trong Series và chỉ mục của hàng có giá trị trong cột 'name' là 'Peter' trong DataFrame.
Hàm index() là một trong những phương pháp quan trọng để truy xuất và xử lý dữ liệu trong Pandas Python.

2.4.6 Hàm replace()

Hàm replace() trong Pandas Python được sử dụng để thay thế giá trị trong một đối tượng Series hoặc DataFrame bằng một giá trị khác.

Cú pháp của hàm replace() như sau:

```
import pandas as pd

# Tạo một Series
s = pd.Series([10, 20, 30, 40, 50])

# Thay thế giá trị 30 bằng giá trị 35 trong Series
s = s.replace(30, 35)

# Tạo một DataFrame
data = {'name': ['John', 'Mary', 'Peter', 'Jane'],
        'age': [25, 30, 35, 40],
        'city': ['New York', 'London', 'Paris', 'Sydney']}
df = pd.DataFrame(data)

# Thay thế tất cả các giá trị trong cột 'city' là 'Paris' bằng giá trị 'Tokyo'
df['city'] = df['city'].replace('Paris', 'Tokyo')
```

Hình 2. 7 Hàm replace()

Trong ví dụ trên, hàm replace() được sử dụng để thay thế giá trị 30 bằng giá trị 35 trong Series và thay thế tất cả các giá trị trong cột 'city' là 'Paris' bằng giá trị 'Tokyo' trong DataFrame.

Hàm replace() cũng có thể được sử dụng để thay thế nhiều giá trị cùng một lúc bằng cách sử dụng một từ điển (dictionary) hoặc list để ánh xạ các giá trị cũ và giá trị mới. Ví dụ:

```
# Thay thế các giá trị 30 và 40 bằng giá trị 35 trong Series
s = s.replace({30: 35, 40: 35})

# Thay thế các giá trị trong cột 'city' là 'Paris' và 'London' bằng giá trị 'Tokyo' và 'Osaka' trong DataFrame
df['city'] = df['city'].replace({'Paris': 'Tokyo', 'London': 'Osaka'})
```

Hình 2. 8 Hàm replace()

Hàm replace() là một trong những phương pháp quan trọng để xử lý dữ liệu trong Pandas Python.

2.4.6 Hàm drop_duplicates()

Hàm drop_duplicates() trong Pandas Python được sử dụng để loại bỏ các hàng trùng lặp trong một đối tượng DataFrame.

Cú pháp của hàm drop_duplicates() như sau:

```python
import pandas as pd

# Tạo một DataFrame
data = {'name': ['John', 'Mary', 'Peter', 'Jane', 'Mary', 'Peter'],
        'age': [25, 30, 35, 40, 30, 35],
        'city': ['New York', 'London', 'Paris', 'Sydney', 'London', 'Paris']}
df = pd.DataFrame(data)

# Loại bỏ các hàng trùng lặp trong DataFrame
df = df.drop_duplicates()
```

Hình 2. 9 Hàm drop_duplicates()

Trong ví dụ trên, hàm drop_duplicates() được sử dụng để loại bỏ các hàng trùng lặp trong DataFrame.
Hàm drop_duplicates() cũng có thể được sử dụng để loại bỏ các hàng trùng lặp dựa trên một hoặc nhiều cột cụ thể bằng cách chỉ định danh sách các tên cột. Ví dụ:

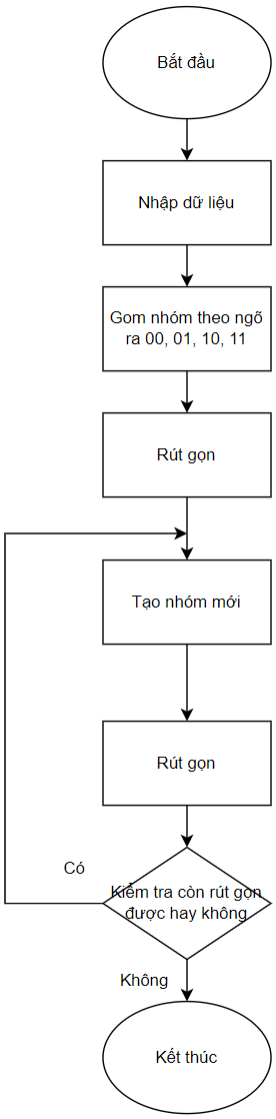
```python
# Loại bỏ các hàng trùng lặp dựa trên cột 'name' và 'age'
df = df.drop_duplicates(subset=['name', 'age'])
```

Hình 2. 10 Hàm drop_duplicates()

Hàm drop_duplicates() là một trong những phương pháp quan trọng để xử lý dữ liệu trong Pandas Python và giúp cho việc phân tích dữ liệu chính xác hơn.

CHƯƠNG 3: THUẬT TOÁN VÀ CÁC CHƯƠNG TRÌNH

3.1 Lưu đồ thuật toán



Hình 3. 1 Lưu đồ thuật toán

3.2 Mô phỏng bước 1 gom nhóm theo ngõ ra

Giả sử ta có bảng dữ liệu sau:

	TTHT	TTKT0	TTKT1	Ngõ ra 0	Ngõ ra 1
0	S0	S0	S4	1	0
1	S1	S1	S5	0	0
2	S2	S0	S4	0	0
3	S3	S1	S5	0	0
4	S4	S3	S6	0	1
5	S5	S3	S6	0	1
6	S6	S2	S7	0	1
7	S7	S2	S7	0	1

Hình 3. 2 Bảng dữ liệu tham khảo

Cách mô phỏng lại thuật toán

Bước 1: gom nhóm theo ngõ ra 00, 01, 10, 11

- Thực hiện bằng tay:

Ta có các nhóm sau:

G0 = {S1, S2, S3}

G1 = {S4, S5, S6, S7}

G2 = {S0}

G3 = {}

- Thực hiện bằng code:

```
In [29]:
import pandas as pd
DF = pd.read_excel(r'D:\4th Semester\CE118 - Thiet ke luan ly so\Do an\TestCase1.xlsx')
G0 = DF[(DF["Ngõ ra 0"] == 0) & (DF["Ngõ ra 1"] == 0)]["TTHT"].tolist()
G1 = DF[(DF["Ngõ ra 0"] == 0) & (DF["Ngõ ra 1"] == 1)]["TTHT"].tolist()
G2 = DF[(DF["Ngõ ra 0"] == 1) & (DF["Ngõ ra 1"] == 0)]["TTHT"].tolist()
G3 = DF[(DF["Ngõ ra 0"] == 1) & (DF["Ngõ ra 1"] == 1)]["TTHT"].tolist()
print(G0)
print(G1)
print(G2)
print(G3)

['S1', 'S2', 'S3']
['S4', 'S5', 'S6', 'S7']
['S0']
[]

In [ ]:
```

Hình 3. 3 Kết quả mô phỏng bước 1

3.3 Mô phỏng bước 2 rút gọn

Bước 2: lập bảng rút gọn

- Ta có các nhóm sau:

  G0 = {S1, S2, S3}

  G1 = {S4, S5, S6, S7}

  G2 = {S0}
- Thực hiện thủ công:

A	G0	G1	G2
0	G0, G2, G0	G0, G0, G0, G0	G2
1	G1, G1, G1	G1, G1, G1, G1	G1

Hình 3. 4 Bảng rút gọn thủ công

- Thực hiện bằng máy:

In [30]:

```
import pandas as pd
DF = pd.read_excel(r'D:\4th Semester\CE118 - Thiet ke luan ly so\Do an\TestCase1.xlsx')
G0 = DF[(DF["Ngõ ra 0"] == 0) & (DF["Ngõ ra 1"] == 0)]["TTHT"].tolist()
G1 = DF[(DF["Ngõ ra 0"] == 0) & (DF["Ngõ ra 1"] == 1)]["TTHT"].tolist()
G2 = DF[(DF["Ngõ ra 0"] == 1) & (DF["Ngõ ra 1"] == 0)]["TTHT"].tolist()
G3 = DF[(DF["Ngõ ra 0"] == 1) & (DF["Ngõ ra 1"] == 1)]["TTHT"].tolist()

DF = DF.replace(G0, 'G0')
DF = DF.replace(G1, 'G1')
DF = DF.replace(G2, 'G2')
DF = DF.replace(G3, 'G3')
DF
```

Out[30]:

	TTHT	TTKT0	TTKT1	Ngõ ra 0	Ngõ ra 1
0	G2	G2	G1	1	0
1	G0	G0	G1	0	0
2	G0	G2	G1	0	0
3	G0	G0	G1	0	0
4	G1	G0	G1	0	1
5	G1	G0	G1	0	1
6	G1	G0	G1	0	1
7	G1	G0	G1	0	1

In [ ]:

Hình 3. 5 Kết quả mô phỏng bước 2

3.4 Mô phỏng bước 3 tạo nhóm mới

Bước 3: Tạo nhóm mới

Từ bảng dữ liệu ở bước 2 ta tạo được nhóm mới như sau:

- Chạy bằng tay:

G0 = {S0}

G1 = {S1, S3}

G2 = {S2}

G3 = {S4, S5, S6, S7}

- Chạy bằng máy: duyệt tuần tự lần lượt xem các hàng có giống nhau hay không để gom nhóm

```
In [2]:
import pandas as pd
df = pd.read_excel(r'D:\4th Semester\CE118 - Thiet ke luan ly so\Do an\TestCase1.xlsx')
DF = df
#print(DF)
G0 = DF[(DF["Ngõ ra 0"] == 0) & (DF["Ngõ ra 1"] == 0)]["TTHT"].tolist()
G1 = DF[(DF["Ngõ ra 0"] == 0) & (DF["Ngõ ra 1"] == 1)]["TTHT"].tolist()
G2 = DF[(DF["Ngõ ra 0"] == 1) & (DF["Ngõ ra 1"] == 0)]["TTHT"].tolist()
G3 = DF[(DF["Ngõ ra 0"] == 1) & (DF["Ngõ ra 1"] == 1)]["TTHT"].tolist()
DF = DF.replace(G0, 'G0')
DF = DF.replace(G1, 'G1')
DF = DF.replace(G2, 'G2')
DF = DF.replace(G3, 'G3')
DF1 = DF
DF = DF.drop_duplicates()
Si = DF.index.tolist()
idx = DF1.index.tolist()
ls = []
for i in Si:
    gr = []
    for j in idx:
        if (len(DF1.iloc[[i, j]].drop_duplicates()) == 1):
            gr.append('S' + str(j))
    ls.append(gr)
print(ls)

[['S0'], ['S1', 'S3'], ['S2'], ['S4', 'S5', 'S6', 'S7']]

In [ ]:
```

Hình 3. 6 Kết quả mô phỏng bước 3

3.5 Mô phỏng bước 4 lặp lại cho đến khi không còn rút gọn được nữa

Chạy bằng tay: quan sát bằng mắt

Chạy bằng máy: kiểm tra xem cột TTHT có còn trùng nhau sau mỗi lần rút gọn hay không

```
import pandas as pd
df = pd.read_excel(r'D:\4th Semester\CE118 - Thiet ke luan ly so\Do an\TestCase2.xlsx')
DF = df
print(DF)
G0 = DF[(DF["Ngõ ra 0"] == 0) & (DF["Ngõ ra 1"] == 0)]["TTHT"].tolist()
G1 = DF[(DF["Ngõ ra 0"] == 0) & (DF["Ngõ ra 1"] == 1)]["TTHT"].tolist()
G2 = DF[(DF["Ngõ ra 0"] == 1) & (DF["Ngõ ra 1"] == 0)]["TTHT"].tolist()
G3 = DF[(DF["Ngõ ra 0"] == 1) & (DF["Ngõ ra 1"] == 1)]["TTHT"].tolist()
DF = DF.replace(G0, 'G0')
DF = DF.replace(G1, 'G1')
DF = DF.replace(G2, 'G2')
DF = DF.replace(G3, 'G3')
DF1 = DF
DF = DF.drop_duplicates()

while (len(DF["TTHT"].tolist()) != len(set(DF["TTHT"].tolist()))):
    Si = DF.index.tolist()
    idx = DF1.index.tolist()
    ls = []
    for i in Si:
        gr = []
        for j in idx:
            if (len(DF1.iloc[[i, j]].drop_duplicates()) == 1):
                gr.append('S' + str(j))
        ls.append(gr)
    DF = df
    for i in ls:
        DF = DF.replace(i, i[0])
    DF1 = DF
    DF = DF.drop_duplicates()
DF = DF.replace({'G0':'S0', 'G1':'S1', 'G2':'S2', 'G3':'S3'})
print(DF)
```

Hình 3. 7 Chương trình hoàn chỉnh

```
  TTHT TTKT0 TTKT1  Ngõ ra 0  Ngõ ra 1
0   S0    S0    S4         1         0
1   S1    S1    S5         0         0
2   S2    S0    S4         0         0
3   S3    S1    S5         0         0
4   S4    S3    S6         0         1
5   S5    S3    S6         0         1
6   S6    S2    S7         0         1
7   S7    S2    S7         0         1
  TTHT TTKT0 TTKT1  Ngõ ra 0  Ngõ ra 1
0   S0    S0    S4         1         0
1   S1    S1    S4         0         0
2   S2    S0    S4         0         0
4   S4    S1    S6         0         1
6   S6    S2    S6         0         1
```

Hình 3. 8 Kết quả sau khi rút gọn

3.6 Chương trình

```
import pandas as pd
df = pd.read_excel(r'D:\4th Semester\CE118 - Thiet ke luan ly so\Do
an\TestCase2.xlsx')
DF = df
print(DF)
G0 = DF[(DF["Ngõ ra 0"] == 0) & (DF["Ngõ ra 1"] == 0)]["TTHT"].tolist()
G1 = DF[(DF["Ngõ ra 0"] == 0) & (DF["Ngõ ra 1"] == 1)]["TTHT"].tolist()
G2 = DF[(DF["Ngõ ra 0"] == 1) & (DF["Ngõ ra 1"] == 0)]["TTHT"].tolist()
G3 = DF[(DF["Ngõ ra 0"] == 1) & (DF["Ngõ ra 1"] == 1)]["TTHT"].tolist()
DF = DF.replace(G0, 'G0')
DF = DF.replace(G1, 'G1')
DF = DF.replace(G2, 'G2')
DF = DF.replace(G3, 'G3')
DF1 = DF
DF = DF.drop_duplicates()

while (len(DF["TTHT"].tolist()) != len(set(DF["TTHT"].tolist()))):
    Si = DF.index.tolist()
    idx = DF1.index.tolist()
    ls = []
    for i in Si:
        gr = []
        for j in idx:
            if (len(DF1.iloc[[i, j]].drop_duplicates()) == 1):
                gr.append('S' + str(j))
```

```
        ls.append(gr)
    DF = df
    for i in ls:
        DF = DF.replace(i, i[0])
    DF1 = DF
    DF = DF.drop_duplicates()
DF = DF.replace({'G0':'S0', 'G1':'S1', 'G2':'S2', 'G3':'S3'})
print(DF)
```

CHƯƠNG 4: THỰC NGHIỆM

4.1 Môi trường thực nghiệm

Thực nghiệm qua 10 testcase với số lượng trạng thái khác nhau

4.2 Kết quả thu được

TestCase1:

```
   TTHT TTKT0 TTKT1  Ngõ ra 0  Ngõ ra 1
0    S0    S0    S4         1         0
1    S1    S1    S5         0         0
2    S2    S0    S4         0         0
3    S3    S1    S5         0         0
4    S4    S3    S6         0         1
5    S5    S3    S6         0         1
6    S6    S2    S7         0         1
7    S7    S2    S7         0         1
```

	TTHT	TTKT0	TTKT1	Ngõ ra 0	Ngõ ra 1
0	S0	S0	S4	1	0
1	S1	S1	S4	0	0
2	S2	S0	S4	0	0
4	S4	S1	S6	0	1
6	S6	S2	S6	0	1

TestCase2:

```
   TTHT TTKT0 TTKT1  Ngõ ra 0  Ngõ ra 1
0    S0    S0    S5         1         0
1    S1    S0    S5         1         0
2    S2    S1    S5         0         0
3    S3    S0    S6         0         0
4    S4    S2    S6         0         0
5    S5    S2    S6         1         1
6    S6    S3    S7         1         1
7    S7    S3    S7         1         1
8    S8    S4    S8         1         1
9    S9    S4    S9         1         1
```

	TTHT	TTKT0	TTKT1	Ngõ ra 0	Ngõ ra 1
0	S0	S0	S5	1	0
2	S2	S0	S5	0	0
4	S4	S2	S5	0	0
5	S5	S2	S5	1	1
8	S8	S4	S8	1	1

TestCase3:

	TTHT	TTKT0	TTKT1	Ngõ ra 0	Ngõ ra 1
0	S0	S0	S1	0	0
1	S1	S2	S1	0	0
2	S2	S0	S1	0	1

	TTHT	TTKT0	TTKT1	Ngõ ra 0	Ngõ ra 1
0	S0	S0	S1	0	0
1	S1	S2	S1	0	0
2	S2	S0	S1	0	1

TestCase4:

	TTHT	TTKT0	TTKT1	Ngõ ra 0	Ngõ ra 1
0	S0	S0	S1	0	0
1	S1	S2	S3	0	0
2	S2	S0	S3	0	0
3	S3	S4	S5	0	1
4	S4	S0	S5	0	1
5	S5	S6	S5	0	1
6	S6	S0	S5	0	1

	TTHT	TTKT0	TTKT1	Ngõ ra 0	Ngõ ra 1
0	S0	S0	S1	0	0
1	S1	S2	S3	0	0
2	S2	S0	S3	0	0
3	S3	S4	S3	0	1
4	S4	S0	S3	0	1

TestCase5:

	TTHT	TTKT0	TTKT1	Ngõ ra 0	Ngõ ra 1
0	S0	S1	S2	0	0
1	S1	S3	S4	0	0
2	S2	S5	S6	0	0
3	S3	S7	S8	0	0
4	S4	S9	S10	0	0
5	S5	S11	S12	0	0
6	S6	S13	S14	0	0
7	S7	S0	S0	0	0
8	S8	S0	S0	0	0
9	S9	S0	S0	0	0
10	S10	S0	S0	1	0
11	S11	S0	S0	0	0
12	S12	S0	S0	1	0
13	S13	S0	S0	0	0
14	S14	S0	S0	0	0

	TTHT	TTKT0	TTKT1	Ngõ ra 0	Ngõ ra 1
0	S0	S1	S2	0	0
1	S1	S3	S4	0	0
2	S2	S4	S3	0	0
3	S3	S7	S7	0	0
4	S4	S7	S10	0	0
7	S7	S0	S0	0	0
10	S10	S0	S0	1	0

TestCase6:

	TTHT	TTKT0	TTKT1	Ngõ ra 0	Ngõ ra 1
0	S0	S1	S5	0	1
1	S1	S2	S3	0	0
2	S2	S0	S2	1	0
3	S3	S1	S5	0	1
4	S4	S2	S3	0	0
5	S5	S0	S4	1	0

	TTHT	TTKT0	TTKT1	Ngõ ra 0	Ngõ ra 1
0	S0	S1	S5	0	1
1	S1	S2	S0	0	0
2	S2	S0	S2	1	0
5	S5	S0	S1	1	0

TestCase7:

	TTHT	TTKT0	TTKT1	Ngõ ra 0	Ngõ ra 1
0	S0	S0	S5	0	1
1	S1	S2	S3	1	0
2	S2	S4	S1	1	1
3	S3	S1	S0	0	1
4	S4	S2	S3	1	0
5	S5	S4	S1	1	1
6	S6	S4	S0	0	1
7	S7	S0	S2	0	1

	TTHT	TTKT0	TTKT1	Ngõ ra 0	Ngõ ra 1
0	S0	S0	S2	0	1
1	S1	S2	S3	1	0
2	S2	S1	S1	1	1
3	S3	S1	S0	0	1

TestCase8:

	TTHT	TTKT0	TTKT1	Ngõ ra 0	Ngõ ra 1
0	S0	S1	S2	0	[illegible]
1	S1	S3	S4	0	[illegible]
2	S2	S5	S6	0	[illegible]
3	S3	S7	S8	0	[illegible]
4	S4	S9	S3	0	[illegible]
5	S5	S2	S11	0	[illegible]
6	S6	S11	S12	0	[illegible]
7	S7	S13	S14	0	[illegible]
8	S8	S0	S0	0	[illegible]
9	S9	S0	S0	0	[illegible]
10	S10	S0	S0	1	[illegible]
11	S11	S0	S0	0	[illegible]
12	S12	S0	S0	1	[illegible]
13	S13	S0	S5	0	[illegible]
14	S14	S0	S0	0	[illegible]

	TTHT	TTKT0	TTKT1	Ngõ ra 0	Ngõ ra 1
0	S0	S1	S2	0	0
1	S1	S3	S4	0	0
2	S2	S5	S6	0	0
3	S3	S7	S8	0	0
4	S4	S8	S3	0	0
5	S5	S2	S8	0	0
6	S6	S8	S10	0	0
7	S7	S13	S8	0	0
8	S8	S0	S0	0	0
10	S10	S0	S0	1	0
13	S13	S0	S5	0	0

TestCase9:

	TTHT	TTKT0	TTKT1	Ngõ ra 0	Ngõ ra 1
0	S0	S11	S1	0	1
1	S1	S12	S3	0	0
2	S2	S10	S2	1	0
3	S3	S11	S5	0	1
4	S4	S12	S3	0	0
5	S5	S10	S4	1	0
6	S6	S16	S5	1	0
7	S7	S0	S5	0	1
8	S8	S1	S5	0	1
9	S9	S2	S5	0	1
10	S10	S3	S5	1	1
11	S11	S5	S5	1	1
12	S12	S5	S5	1	1
13	S13	S5	S5	1	0
14	S14	S5	S5	1	0
15	S15	S5	S5	0	0
16	S16	S5	S5	0	0

	TTHT	TTKT0	TTKT1	Ngõ ra 0	Ngõ ra 1
0	S0	S11	S1	0	1
1	S1	S11	S3	0	0
2	S2	S10	S2	1	0
3	S3	S11	S5	0	1
5	S5	S10	S1	1	0
6	S6	S15	S5	1	0
7	S7	S0	S5	0	1
8	S8	S1	S5	0	1
9	S9	S2	S5	0	1
10	S10	S3	S5	1	1
11	S11	S5	S5	1	1
13	S13	S5	S5	1	0
15	S15	S5	S5	0	0

Tescase 10:

	TTHT	TTKT0	TTKT1	Ngõ ra 0	Ngõ ra 1
0	S0	S1	S5	0	1
1	S1	S2	S3	0	0
2	S2	S0	S2	1	0
3	S3	S1	S5	0	1
4	S4	S2	S3	0	0
5	S5	S0	S4	1	0
6	S6	S1	S10	0	0
7	S7	S2	S2	0	0
8	S8	S2	S3	0	0
9	S9	S2	S5	0	0
10	S10	S2	S7	0	1
11	S11	S3	S9	0	1
12	S12	S3	S9	0	1
13	S13	S4	S9	0	1
14	S14	S5	S10	1	0
15	S15	S6	S2	1	0
16	S16	S7	S2	1	0
17	S17	S2	S2	1	0
18	S18	S3	S2	0	1
19	S19	S2	S2	0	1
20	S20	S2	S2	0	1
21	S21	S3	S3	1	1
22	S22	S3	S4	1	1
23	S23	S3	S5	1	1
24	S24	S3	S6	1	1
25	S25	S3	S7	1	1

	TTHT	TTKT0	TTKT1	Ngõ ra 0	Ngõ ra 1
0	S0	S1	S5	0	1
1	S1	S2	S0	0	0
2	S2	S0	S2	1	0
5	S5	S0	S1	1	0
6	S6	S1	S10	0	0
7	S7	S2	S2	0	0
9	S9	S2	S5	0	0
10	S10	S2	S7	0	1
11	S11	S0	S9	0	1
13	S13	S1	S9	0	1
14	S14	S5	S10	1	0
15	S15	S6	S2	1	0
16	S16	S7	S2	1	0
17	S17	S2	S2	1	0
18	S18	S0	S2	0	1
19	S19	S2	S2	0	1
21	S21	S0	S0	1	1
22	S22	S0	S1	1	1
23	S23	S0	S5	1	1
24	S24	S0	S6	1	1
25	S25	S0	S7	1	1

Present State	Next State	
	A = 0	A = 1
S0	S1/0	S5/1
S1	S2/0	S0/0
S2	S0/1	S2/0
S5	S0/1	S1/0
S6	S1/0	S10/0
S7	S2/0	S2/0
S9	S2/0	S5/0
S10	S2/0	S7/1
S11	S0/0	S9/1
S13	S1/0	S9/1
S14	S5/1	S10/0
S15	S6/1	S2/0
S16	S7/1	S2/0
S17	S2/1	S2/0
S18	S0/0	S2/1
S19	S2/0	S2/1
S21	S0/1	S0/1
S22	S0/1	S1/1
S23	S0/1	S5/1
S24	S0/1	S6/1
S25	S0/1	S7/1

CHƯƠNG 5: ĐÁNH GIÁ

5.1 Độ chính xác

Toàn bộ 10 testcase đã được chạy thủ công và chương trình cho độ chính xác 100% như cách làm thủ công.

5.2 Thời gian đáp ứng

Tùy thuộc vào số lượng trạng thái mà thuật toán sẽ tiêu tốn thời gian, vì thuật toán có độ phức tạp $O(n^2)$, với n là số lượng trạng thái.

5.3 Một số hạn chế còn tồn tại

Do chương trình chỉ được kiểm nghiệm qua những testcase có sẵn và số lượng testcase còn ít nên khó có thể kết luận là chương trình đã chạy đúng hoàn toàn.

5.4 Kết luận

Vì đây là hệ thống được thực hiện trên phần mềm mô phỏng nên so với thực tế sẽ còn nhiều sự khác biệt.

Hướng phát triển: cần những phương pháp và chương trình khác cùng làm về chủ đề này để có thể kiểm định độ chính xác.

TÀI LIỆU THAM KHẢO

https://sparkbyexamples.com/python-pandas-tutorial-for-beginners/