

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA KỸ THUẬT MÁY TÍNH

—○○○○—



ĐỒ ÁN MÔN:
THIẾT KẾ LUẬN LÝ SỐ

Đề tài: Rút gọn và mã hóa bảng trạng thái cho trước

Sinh viên thực hiện:

- 1. Nguyễn Tiến Toàn**
- 2. Bành Trí Kiệt**

MSSV: 21521548
MSSV: 21522251

Lớp:

CE118.N22

Giáo viên hướng dẫn:

Th.S Trương Văn Cường

Lời nói đầu

Ta biết rằng, số lượng trạng thái của một FSM (máy trạng thái hữu hạn) là một con số có thể thay đổi tùy thuộc vào người thiết kế, điều đó cũng có nghĩa rằng ta có thể thiết kế nhiều máy FSM có cùng chức năng nhưng khác nhau về số lượng trạng thái, từ đó dẫn đến sự khác nhau về diện tích, tốc độ, công suất tiêu thụ,...

Vì vậy, rút gọn số lượng trạng thái ban đầu là một bài toán mà người thiết kế phải giải quyết được nếu muốn máy FSM tối ưu về chi phí, việc rút gọn càng nhiều trạng thái, thậm chí đến mức gọi là tối giản (số lượng trạng thái không thể ít hơn) có thể tiết kiệm diện tích, điển hình là số Flip Flop cần được sử dụng hoặc giúp thuận lợi hơn trong việc phát triển mạch.

Để có thể hỗ trợ cũng như một cách kiểm tra kết quả bài toán trên, nhóm xin được giới thiệu một chương trình có thể rút gọn được bảng trạng thái kể cả với số lượng trạng thái ban đầu lớn và có nhiều giá trị ngõ vào.

Bài báo cáo về đồ án phần mềm rút gọn bảng trạng thái được chia thành 4 chương:

- **Chương 1: GIỚI THIỆU ĐỒ ÁN**
- **Chương 2: KIẾN THỨC VỀ NGÔN NGỮ LẬP TRÌNH PYTHON**
- **Chương 3: GIẢI THUẬT CỦA CHƯƠNG TRÌNH**
- **Chương 4: NHẬN XÉT VỀ SẢN PHẨM**

MỤC LỤC



CHƯƠNG 1: GIỚI THIỆU PHẦN MỀM.....	1
1.1. Tổng quan	1
1.2. Hướng dẫn sử dụng.....	1
CHƯƠNG 2: KIẾN THỨC VỀ NGÔN NGỮ LẬP TRÌNH PYTHON.....	2
2.1. Giới thiệu	2
2.2. Tìm hiểu ngôn ngữ lập trình Python.....	2
2.2.1. Khái quát ngôn ngữ lập trình Python.....	2
2.2.2 Thư viện Pandas.....	3
2.2.3 Thư viện Xlwings và thư viện Openpyxl	6
2.2.4 Các hàm (phương thức) hỗ trợ:	6
CHƯƠNG 3: GIẢI THUẬT CỦA CHƯƠNG TRÌNH.....	9
3.1. Rút gọn bảng trạng thái.....	9
3.2. Mã hóa trạng thái	12
3.2.1. Mã hóa theo số nhị phân:.....	12
3.2.2. Mã hóa theo hot-one	13
3.3 Xuất ra file kết quả.....	14
CHƯƠNG 4: NHẬN XÉT VỀ SẢN PHẨM	15
4.1. Những điểm cần lưu ý.....	15
4.2. Nhận xét về chương trình	16

DANH MỤC HÌNH ẢNH

Hình 1-1	Bảng trạng thái ban đầu.....	1
Hình 2-1	Một file lập trình Python	2
Hình 2-2	Câu lệnh khai báo thư viện, đọc, in file Excel	4
Hình 2-3	Câu lệnh truy vấn dữ liệu trong file Excel	5
Hình 2-4	Câu lệnh xóa hàng và cột trong file Excel	5
Hình 2-5	Hình ảnh câu lệnh xóa khoảng trắng.....	6
Hình 2-6	Hình ảnh câu lệnh thay thế kí tự trong chuỗi	7
Hình 2-7	Hình ảnh câu lệnh đổi số thập phân thành nhị phân	7
Hình 2-8	Hình ảnh câu lệnh dịch trái 1 bit	8
Hình 3-1	Bảng trạng thái ban đầu.....	9
Hình 3-2	bảng trạng thái sau khi rút gọn.....	11
Hình 3-3	Bảng trạng thái được mã hóa theo số đếm nhị phân	12
Hình 3-4	Bảng trạng thái được mã hóa theo hot-one	13
Hình 3-5	Hình ảnh bảng trạng thái trong file “Output.xlsx”	14
Hình 4-1	Bảng trạng thái không hợp lệ	15

CHƯƠNG 1: GIỚI THIỆU PHẦN MỀM

1.1. Tổng quan

Đây là một phần mềm giúp người dùng rút gọn số lượng trạng thái ban đầu xuống mức tối thiểu, ngoài ra còn có thể hỗ trợ việc mã hóa trạng thái theo 2 phương pháp là số nhị phân và mã hot-one.

Người dùng có thể sử dụng phần mềm để hỗ trợ trong quá trình học tập và thiết kế mạch một cách dễ dàng hơn khi đã có thể

1.2. Hướng dẫn sử dụng

Người dùng cần nhập bảng trạng thái cần rút gọn vào một file excel rồi lưu file lại, sau đó đưa đường dẫn chứa địa chỉ file đó vào file **.ipynb** đã được lập trình để rút gọn và mã hóa. Kết quả sau đó được xuất ra trong terminal của chương trình. Sau khi thực hiện chương trình sẽ tạo một file Excel có tên là Output.xlsx chứa bảng có số lượng trạng thái là tối thiểu và có thêm một cột mã hóa trạng thái (nếu người dùng có lựa chọn chức năng mã hóa).

	A	B	C	D	E	F
1	TTHT	TTKT0	TTKT1	Ngõ ra 0	Ngõ ra 1	
2	S0	S0	S4	1	0	
3	S1	S0	S4	0	0	
4	S2	S1	S5	0	0	
5	S3	S1	S5	0	0	
6	S4	S2	S6	0	1	
7	S5	S2	S6	0	1	
8	S6	S3	S7	0	1	
9	S7	S3	S7	0	1	
10						

Hình 1-1 Bảng trạng thái ban đầu

CHƯƠNG 2: KIẾN THỨC VỀ NGÔN NGỮ LẬP TRÌNH PYTHON

2.1. Giới thiệu

Để thiết kế được phần mềm rút gọn bảng trạng thái, nhóm đã sử dụng ngôn ngữ lập trình Python trong đó có sự hỗ trợ của 2 thư viện: Xlwings và Pandas để hiện thực giải thuật nhóm đã đề ra.

Vì vậy, để hiểu rõ quy trình vận hành cũng như hiện thực ý tưởng lên trên trình biên dịch, người thiết kế cần có kiến thức về ngôn ngữ lập trình Python, thư viện Pandas, Xlwings và kỹ năng rút gọn bảng trạng thái đã được học môn bộ môn Thiết kế luận lý số.

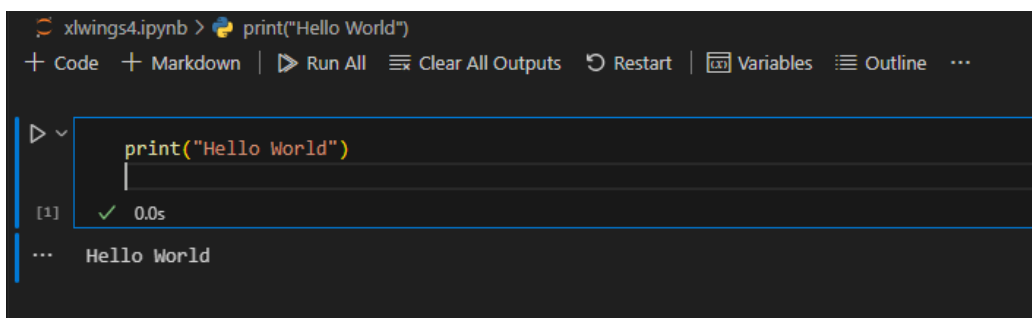
Sau đây nhóm sẽ khái quát sơ lược về những kiến thức trọng tâm, những hàm đặc trưng của từng thư viện để từ đó hiểu được lý do cũng như vai trò của chúng trong việc rút gọn và mã hóa trạng thái.

2.2. Tìm hiểu ngôn ngữ lập trình Python

2.2.1. Khái quát ngôn ngữ lập trình Python

Python là ngôn ngữ lập trình máy tính bậc cao bởi nó được thiết kế để *giúp người học dễ đọc, dễ hiểu và dễ nhớ*; vì thế ngôn ngữ Python có hình thức rất sáng sủa, cấu trúc rõ ràng, thuận tiện cho người mới học. Nói cách khác thì so với các ngôn ngữ lập trình khác, chúng ta có thể sử dụng ít dòng code hơn để viết ra một chương trình trong Python.

Thông thường, khi lập trình bằng ngôn ngữ python, lập trình viên có lưu những đoạn code trong file có đuôi là .py; .ipynb, ... Và nhóm đã lưu file code của mình dưới dạng đuôi .ipynb, dựa trên nền tảng Jupyter Notebook bởi đây là mã nguồn mở, sẽ hỗ trợ thuận tiện hơn trong việc truy xuất và nhập dữ liệu từ file Excel (nơi chứa bảng trạng thái trước và sau khi rút gọn và mã hóa).

The image shows a Jupyter Notebook window titled 'xlwings4.ipynb'. The top bar contains icons for Code, Markdown, Run All, Clear All Outputs, Restart, Variables, and Outline. The main area shows a code cell with the text 'print("Hello World")'. Below the code cell, there is a status bar indicating '[1] ✓ 0.0s'. At the bottom, the output of the code is displayed as 'Hello World'.

Hình 2-1 Một file lập trình Python

2.2.2 Thư viện Pandas

Pandas (viết tắt từ Panel Data - bảng dữ liệu) là thư viện mã nguồn mở phục vụ cho việc phân tích và xử lý dữ liệu trong Python. Thư viện này được thiết kế để làm việc dễ dàng và trực quan với dữ liệu có cấu trúc (dạng bảng, đa chiều, ...) và dữ liệu chuỗi thời gian. Hiện nay, Pandas được sử dụng rộng rãi trong cả nghiên cứu lẫn phát triển các ứng dụng về khoa học dữ liệu.

Thư viện Pandas chính là yếu tố quan trọng nhất để hiện thực phần mềm rút gọn này. Và sau đây ta sẽ tìm hiểu các câu lệnh và hàm thuộc Thư viện Pandas mà nhóm đã sử dụng cho phân lập trình phần mềm.

- **import pandas as pd**

Đây là câu lệnh dùng để khai báo thư viện pandas, là câu lệnh bắt buộc phải có nếu muốn sử dụng các hàm trong một thư viện.

- **f1 = pd.read_excel("DUONG_DAN")**

Để đọc 1 file excel, ta đưa đường dẫn vị trí của file đó vào phần "DUONG_DAN".

Ví dụ DUONG_DAN = 'C:/Users/tient/OneDrive/Máy tính/Project/Do_An_Luan_Ly_So/Vo_Phu_Toan/Test.xlsx'

Ngoài ra, trong câu lệnh trên, ta đã gán trang tính cần đọc vào biến f1, vì vậy mỗi khi tương tác với file excel, ta sẽ thao tác thông qua biến f1.

- **print(f1)**

Đây là câu lệnh dùng để in thông tin có trong file Excel.

- **print(f1.iloc[3,1])**

Hàm iloc[rows, columns]: dùng để truy xuất đến giá trị nằm tại hàng (rows), cột (columns). Theo ví dụ là in giá trị tại vị trí hàng 3 cột 1.

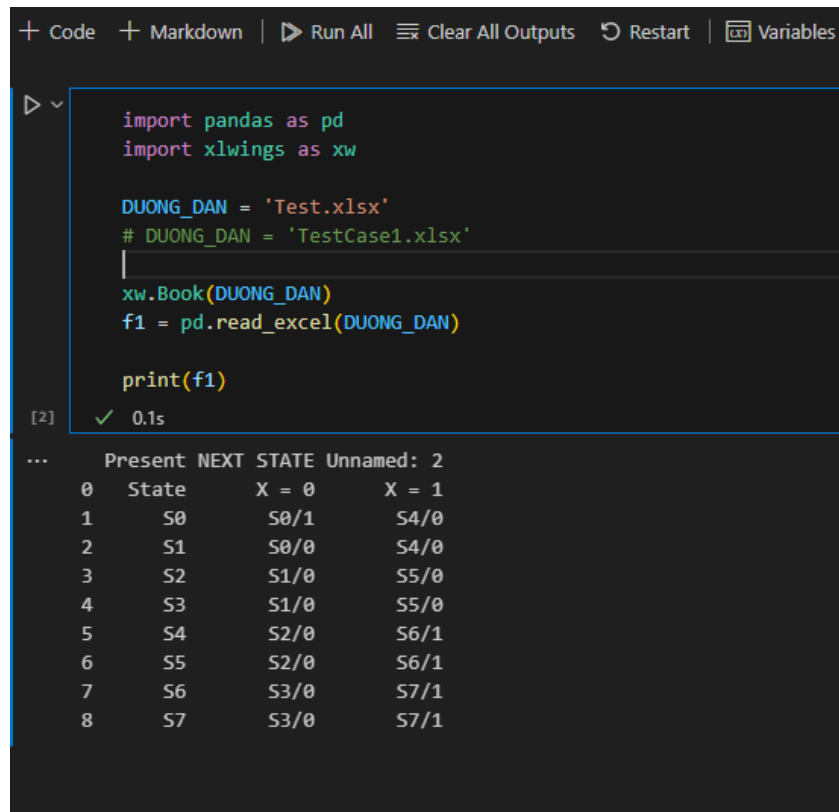
- **f1 = f1.drop(i)**

Đây là câu lệnh dùng để xóa hàng trong bảng, cụ thể là xóa hàng thứ i.

- **f1 = f1.drop(columns [i], axis = 1):**

Đây là câu lệnh dùng để xóa cột trong trang tính, ở đây là xóa cột thứ i.

Hình ảnh minh họa cho các câu lệnh đã giới thiệu ở trên:



```
+ Code + Markdown | ▶ Run All ☰ Clear All Outputs ↺ Restart | Variables
```

```
import pandas as pd
import xlwings as xw

DUONG_DAN = 'Test.xlsx'
# DUONG_DAN = 'TestCase1.xlsx'
|
xw.Book(DUONG_DAN)
f1 = pd.read_excel(DUONG_DAN)

print(f1)
```

[2] ✓ 0.1s

... Present NEXT STATE Unnamed: 2

0	State	X = 0	X = 1
1	S0	S0/1	S4/0
2	S1	S0/0	S4/0
3	S2	S1/0	S5/0
4	S3	S1/0	S5/0
5	S4	S2/0	S6/1
6	S5	S2/0	S6/1
7	S6	S3/0	S7/1
8	S7	S3/0	S7/1

Hình 2-2 Câu lệnh khai báo thư viện, đọc, in file Excel


```

import pandas as pd
import xlwings as xw

DUONG_DAN = 'Test.xlsx'
# DUONG_DAN = 'TestCase1.xlsx'

xw.Book(DUONG_DAN)
f1 = pd.read_excel(DUONG_DAN)

print(f1, '\n')

print(f1.iloc[3,1])

```

✓ 0.6s

	Present	NEXT	STATE	Unnamed: 2
0	State	X = 0	X = 1	
1	S0	S0/1	S4/0	
2	S1	S0/0	S4/0	
3	S2	S1/0	S5/0	
4	S3	S1/0	S5/0	
5	S4	S2/0	S6/1	
6	S5	S2/0	S6/1	
7	S6	S3/0	S7/1	
8	S7	S3/0	S7/1	

S1/0

Hình 2-3 Câu lệnh truy vấn dữ liệu trong file Excel

```

f1 = f1.drop(3)
f1 = f1.drop(f1.columns[2], axis = 1)

print(f1)

```

[19] ✓ 2.2s

	Present	NEXT	STATE
0	State	X = 0	
1	S0	S0/1	
2	S1	S0/0	
4	S3	S1/0	
5	S4	S2/0	
6	S5	S2/0	
7	S6	S3/0	
8	S7	S3/0	

Hình 2-4 Câu lệnh xóa hàng và cột trong file Excel

2.2.3 Thư viện Xlwings và thư viện Openpyxl

Bên cạnh thư viện Pandas, thư viện Xlwings và thư viện Openpyxl cũng là các thư viện Python dùng để tương tác với Microsoft Excel. Thậm chí thư viện Xlwings còn tích hợp thư viện pandas, hay nói cách khác ta có thể sử dụng thư viện này để thực hiện các chức năng truy vấn, thay đổi dữ liệu vừa được giới thiệu ở trên.

Tuy vậy, trong khuôn khổ đề án, nhóm chú trọng vào thư viện Pandas bởi các hàm thực hiện yêu cầu truy vấn hay chỉnh sửa dữ liệu có cú pháp ngắn gọn hơn. Còn 2 thư viện này chúng ta chỉ sử dụng vài câu lệnh đơn giản chứ không tìm hiểu kỹ các hàm đặc trưng.

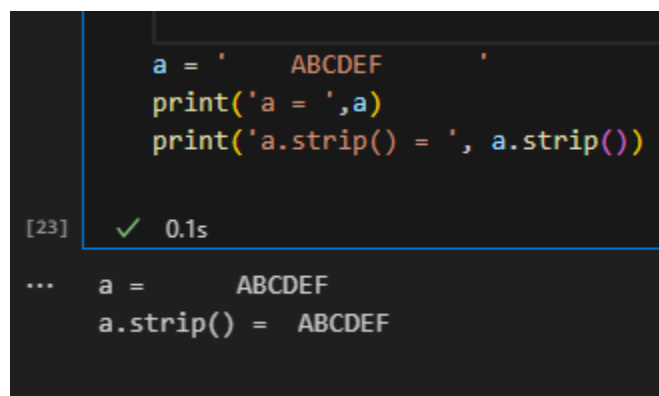
Trong đó ta sử dụng thư viện Xlwings để giúp tự động mở file excel bảng trạng thái ban đầu và file excel chứa bảng trạng thái sau khi đã biến đổi, còn thư viện Openpyxl được dùng để lược bỏ các giá trị thừa trong file excel chứa bảng trạng thái kết quả để hỗ trợ người xem có thể quan sát thuận tiện hơn

2.2.4 Các hàm (phương thức) bổ trợ

Bên cạnh các hàm thuộc 2 thư viện Pandas và Xlwings ở trên, chúng ta cũng sẽ sử dụng các hàm xử lý chuỗi đã được tích hợp sẵn trong Python và có thể gọi ra mà không cần thông qua bất cứ thư viện nào, giúp thuận tiện hơn trong việc xử lý chuỗi thứ cũng chính là điểm mạnh của Python khi so sánh với các ngôn ngữ lập trình khác.

- **Hàm: `rstrip(chuoi)`**

Đây là hàm dùng để xóa khoảng trắng trước và sau chuỗi ký tự.



```
a = '  ABCDEF  '
print('a = ',a)
print('a.strip() = ', a.strip())

[23] ✓ 0.1s

... a =      ABCDEF
    a.strip() = ABCDEF
```

Hình 2-5 Hình ảnh câu lệnh xóa khoảng trắng

- **Hàm: .replace(a, b, count)**

Vai trò của của hàm này là để thay đổi kí tự ('a') trong chuỗi đã chọn thành kí tự ('b'), tối đa là (count) lần.

```
a = 'S3/1'
b = a.replace(a[0:2], 'S1', 1)

print ('a = ',a)
print ('b = ',b)
```

[25] ✓ 0.1s

```
... a = S3/1
    b = S1/1
```

Hình 2-6 Hình ảnh câu lệnh thay thế kí tự trong chuỗi

- **Hàm: getbinary(n, m)**

Giúp đổi số thập phân n sang nhị phân với độ dài (m) bit

```
getbinary = lambda x, n: format(x, 'b').zfill(n)
a = getbinary(5, 4)
print(a)
```

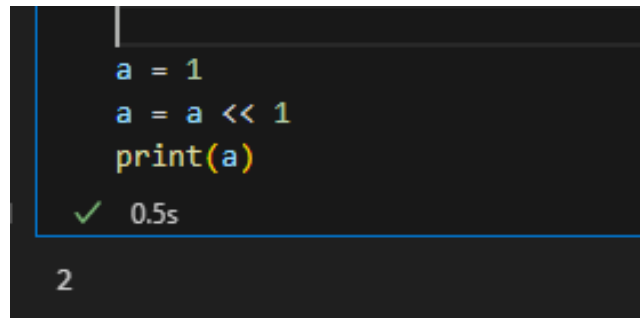
1 ✓ 24.3s

```
0101
```

Hình 2-7 Hình ảnh câu lệnh đổi số thập phân thành nhị phân

- **Hàm: << 1**

Dịch trái giá trị tham số 1 bit, bản chất là nhân số đó với 2.



```
a = 1
a = a << 1
print(a)
```

✓ 0.5s

2

Hình 2-8 Hình ảnh câu lệnh dịch trái 1 bit

CHƯƠNG 3: GIẢI THUẬT CỦA CHƯƠNG TRÌNH

3.1. Rút gọn bảng trạng thái

Quan sát bảng trạng thái dưới đây, ta thấy một bảng trạng thái ban đầu chưa cần được rút gọn và mã hóa nếu muốn.

	A	B	C	D
1	Present	NEXT STATE		
2	State	X = 0	X = 1	
3	S0	S0/1	S4/0	
4	S1	S0/0	S4/0	
5	S2	S1/0	S5/0	
6	S3	S1/0	S5/0	
7	S4	S2/0	S6/1	
8	S5	S2/0	S6/1	
9	S6	S3/0	S7/1	
10	S7	S3/0	S7/1	
11				

Hình 3-1 Bảng trạng thái ban đầu

Nếu đã học môn thiết kế luận lý số, ta biết rằng sẽ có 2 cách để ta có thể tự rút gọn số lượng trạng thái, tuy vậy khi thiết kế phần mềm, nhóm đã sử dụng một cách khác thay vì 2 phương pháp trên.

Ta có định nghĩa rằng: Hai trạng thái được coi là tương đương nếu chúng có cùng trạng thái kế tiếp và giá trị ngõ ra ứng với mỗi giá trị ngõ vào. Nhìn vào bảng trạng thái trên, ta có thể hiểu rằng 2 trạng thái là tương đương nếu chúng chỉ khác nhau giá trị tại cột “TTHT”, còn các cột còn lại (“TTKT0”, “TTKT1”, “Ngõ ra 0”, “Ngõ ra 1”) thì đều có giá trị giống nhau.

Vì vậy, công việc rút gọn của chúng ta bây giờ là: quan sát hàng **i** và **j** ($i < j$), nếu giá trị tại các cột so sánh **TTKT_x** và **Ngõ ra y** (x, y có thể là 0 hoặc 1) thì ta có thể kết luận trạng thái tại hàng **j** tương đương với trạng thái ở hàng **i**.

Ví dụ quan sát hình 3-1, ta thấy 2 trạng thái “S0” và “S5” tương đương, bởi chúng có chung giá trị tại các ô **TTKT**, **Ngõ ra** tương đương với định nghĩa là giống nhau trạng thái kế tiếp và giá trị ngõ ra ứng với mỗi giá trị ngõ vào.

Sau khi ta tìm được các cặp trạng thái tương đương , việc tiếp theo ta sẽ làm là xóa 1 trong 2 khỏi bảng trạng thái, sau đó thay thế vị trí của trạng thái đã bị loại bỏ bằng trạng thái còn lại ở các ô thuộc TTKT.

Ví dụ, khi thấy “S2” và “S3” tương đương nhau, ta sẽ dòng chứa TTHT là “S3” ở hàng 6, sau đó ta sẽ thay thế tất cả giá trị “S3” (vị trí B9, B10) bằng “S2”.

Lặp lại quy trình này đến khi số dòng của bảng không còn thay đổi khi ta rút gọn bảng trạng thái thì dừng lại, lúc này ta đã có cho mình bảng có số lượng trạng thái đã được rút gọn tối giản.

Trong các bước thực hiện ở trên, ta đã sử dụng 3 hàm để thực hiện công việc đó là:

- Hàm **iloc[rows, columns]**
Dùng để truy vấn dữ liệu để so sánh.
- Hàm **f1 = f1.drop(i)**
Dùng để xóa bỏ hàng chứa trạng thái tương đương
- Hàm **replace(a, b, count)**
Thay đổi giá trị tại ô nhớ trong bảng sau khi rút gọn trạng thái.

- Bảng trạng thái sau khi rút gọn sẽ được xuất ra terminal để người dùng quan sát và lựa chọn có mã hóa trạng thái hay không.

... Bảng trạng thái ban đầu:

	Present	NEXT	STATE	Unnamed: 2
0	State		X = 0	X = 1
1	S0		S0/1	S4/0
2	S1		S0/0	S4/0
3	S2		S1/0	S5/0
4	S3		S1/0	S5/0
5	S4		S2/0	S6/1
6	S5		S2/0	S6/1
7	S6		S3/0	S7/1
8	S7		S3/0	S7/1

Bảng trạng thái sau khi rút gọn:

	Present	NEXT	STATE	Unnamed: 2
0	State		X = 0	X = 1
1	S0		S0/1	S4/0
2	S1		S0/0	S4/0
3	S2		S1/0	S4/0
5	S4		S2/0	S4/1

Hình 3-2 bảng trạng thái sau khi rút gọn

3.2. Mã hóa trạng thái

3.2.1. Mã hóa theo số nhị phân:

Mã hóa theo số đếm được hiểu là mã hóa theo thứ tự số đếm nhị phân tăng dần. Vì vậy bảng trạng thái có n trạng thái thì sẽ được mã hóa theo nhị phân ứng với giá trị từ $(0 \rightarrow n-1)$

Để thực hiện việc mã hóa này, đầu tiên ta cần kiểm tra xem bảng (sau khi đã rút gọn) có bao nhiêu trạng thái. Sau đó ta mã hóa nhị phân từng trạng thái ứng với số thứ tự của chúng từ trên xuống dưới.

Ví dụ trạng thái ở vị trí thứ 2 từ trên xuống sẽ được mã hóa là “10” (do số “2” thập phân đổi thành nhị phân là “10”).

```
...
    Bạn muốn mã hóa theo:
    (1): Mã hóa nhị phân
    (2): Mã hóa hot one
    Có số trạng thái là: 4
    --> Vì vậy số bit cần: 2

    Bảng trạng thái sau khi mã hóa:
    Present NEXT STATE Unnamed: 2 Mã hóa Nhị Phan
    0   State      X = 0      X = 1
    1     S0        S0/1        S4/0        00
    2     S1        S0/0        S4/0        01
    3     S2        S1/0        S4/0        10
    5     S4        S2/0        S4/1        11

    ----- Kết thúc chương trình -----
```

Hình 3-3 Bảng trạng thái được mã hóa theo số đếm nhị phân

3.2.2. Mã hóa theo hot-one

Mã hot one sử dụng 1 bit trạng thái để gán cho một trạng thái, nói cách khác, mỗi trạng thái được phân biệt với các trạng thái khác nhờ bit trạng thái dành cho nó có giá trị 1 trong khi các bit trạng thái còn lại bằng 0. Theo đó nếu FSM có n trạng thái thì cần n bit trạng thái để mã hóa theo hot one. Như vậy mỗi mã nhị phân được gán cho 1 trạng thái chỉ có duy nhất 1 bit 1 mà thôi.

- Phương pháp mà phần mềm sử dụng để mã hóa theo hot one:

- Trạng thái đầu tiên được mã hóa là 1
- Trạng thái thứ 2 sẽ được mã hóa là $1 \ll 1$
- Trạng thái thứ n sẽ được mã hóa là $1 \ll n-1$

```
[3] ✓ 1m 37.7s
...
  Bạn muốn mã hóa theo:
  (1): Mã hóa nhị phân
  (2): Mã hóa hot one
  Bảng trạng thái sau khi mã hóa:
    Present NEXT STATE Unnamed: 2 Mã hóa hot-one
  0   State      X = 0      X = 1
  1   S0         S0/1       S4/0       0001
  2   S1         S0/0       S4/0       0010
  3   S2         S1/0       S4/0       0100
  5   S4         S2/0       S4/1       1000

  ----- Kết thúc chương trình -----
```

Hình 3-4 Bảng trạng thái được mã hóa theo hot-one

3.3 Xuất ra file kết quả

Sau khi đã thực hiện xong quá trình rút gọn và mã hóa (nếu có), chương trình sẽ xuất ra file có tên là “Output.xlsx” ở vị trí nằm chung với thư mục chứa phần mềm.

Sau đó, chương trình tiếp tục chỉnh sửa bố cục của file “Output.xlsx” bằng cách xóa đi các ô có chữ “Unname...” trong file kết quả (có thể quan sát hình 3.2 hoặc 3.3) có thể xuất hiện nếu bảng trạng thái ban đầu có sử dụng tính năng gộp ô (Merge). Ta sẽ dùng thư viện Openpyxl để thực hiện công việc này

Sau đó dùng hàm mở file tự động “xw.Book()” của thư viện Xlwings để máy tự động mở file excel “Output.xlsx” để người dùng quan sát kết quả sau cùng của chương trình.

	A	B	C	D	E
1	Present	NEXT STATE	Mã	hóa	Nhi Phan
2	State	X = 0	X = 1		
3	S0	S0/1	S4/0	00	
4	S1	S0/0	S4/0	01	
5	S2	S1/0	S4/0	10	
6	S4	S2/0	S4/1	11	
7					

Hình 3.5 – Hình ảnh bảng trạng thái trong file “Output.xlsx”

CHƯƠNG 4: NHẬN XÉT VỀ SẢN PHẨM

4.1. Những điểm cần lưu ý

Một ưu điểm của phần mềm đó là vẫn có thể hoạt động một cách bình thường cho dù bảng trạng thái ở vị trí bất kỳ trong file excel.

Tuy nhiên, để phần mềm có thể hoạt động chính xác thì file excel đòi hỏi không được có những ký tự thừa bên trong hoặc ngoài bảng hoặc bảng bị tách cột, dòng làm mất định dạng ban đầu của bảng trạng thái. Như vậy sẽ khiến chương trình chạy sai.

			fghfg			
		Present	NEXT STATE			
		State	X = 0	X = 1		
		S0	S0/1	S4/0		
		S1	S0/0	S4/0		
		S2	S1/0	S5/0		
		S3	S1/0	S5/0		
â		S4	S2 /0	S6/1		fhfgf gg
		S5	S2/0	S6/1		
		S6	S3/0	S7/1		
		S7	S3/0	S7/1		
			bvbv			

Hình 4-1 Bảng trạng thái không hợp lệ

4.2. Nhận xét về chương trình

Nhóm chỉ đào sâu chủ yếu về thư viện Pandas chứ chưa khai thác hết nguồn kiến thức về 2 thư viện cũng vô cùng mạnh mẽ về giao tiếp với file Excel đó là Xlwings và openpyxl, điều có thể giúp cho file kết quả “Output.xlsx” trông đẹp mắt hơn.

Chương trình vẫn chưa có tính năng mã hóa trạng thái theo phương pháp rút gọn ưu tiên liên kề, một phương pháp vô cùng hữu hiệu đối với những bảng trạng thái có số lượng trạng thái lớn và nhiều giá trị ngõ vào.