

Môn: **Nhập môn Trí tuệ nhân tạo**

Nhóm: **6**

Giảng viên: **Nguyễn Thành An**

Mã nhóm: **PBL**

#	Họ và tên	MSSV	Email
1	Võ Phước Thịnh	52000807	
2	Nguyễn Trung Nghĩa	52000693	nguyentrungnghiacmp528@gmail.com
3	Hồ Thịnh Phát	51900821	hothinhphatvd4@gmail.com
4	Trần Đình Phúc	52000109	phuctran12421@gmail.com

---

## BẢNG PHÂN CÔNG CÔNG VIỆC

#	Công việc	Người thực hiện	Đánh giá
1	Câu 1	Nguyễn Trung Nghĩa	Hoàn thành
2	Câu 2	Trần Đình Phúc	Hoàn thành
3	Câu 3	Nguyễn Trung Nghĩa	Hoàn thành
4	Câu 4	Hồ Thịnh Phát	Hoàn thành
5	Câu 5	Hồ Thịnh Phát	Hoàn thành
6	Kiểm thử, chỉnh sửa lại	Trần Đình Phúc	Hoàn thành

#	Họ và tên	MSSV	% Đánh giá
1	Võ Phước Thịnh	52000807	0
2	Nguyễn Trung Nghĩa	52000693	100
3	Hồ Thịnh Phát	51900821	0
4	Trần Đình Phúc	52000109	100

Workspace: <https://github.com/trngbro/AI> (Private during deadline time)

Câu 1 .....	4
YC1.1 .....	6
YC1.2 .....	7
Câu 2 .....	8
YC2.1 .....	9
YC2.2 .....	10
YC2.3 .....	11
Câu 3 .....	12
YC3.1 .....	13
YC3.2 .....	14
Câu 4 .....	15
YC4.1 .....	16
YC4.2 .....	17
YC4.3 .....	18
YC4.4 .....	19
Tổng kết .....	20
Thuận lợi & Khó khăn .....	21 & 22

# **Câu 1 - Constraint Satisfaction**

# Backtracking search

```
function BACKTRACKING-SEARCH(csp) returns a solution, or failure
  return RECURSIVE-BACKTRACKING({}, csp)

function RECURSIVE-BACKTRACKING(assignment, csp) returns a solution, or failure
  if assignment is complete then return assignment
  var ← SELECT-UNASSIGNED-VARIABLE(Variables[csp], assignment, csp)
  for each value in ORDER-DOMAIN-VALUES(var, assignment, csp) do
    if value is consistent with assignment according to Constraints[csp] then
      add { var = value } to assignment
      result ← RECURSIVE-BACKTRACKING(assignment, csp)
      if result ≠ failure then return result
      remove { var = value } from assignment
  return failure
```

*Tham khảo giải thuật Backtracking trong slide bài giảng lesson 05*

## Câu 1 (2.0 điểm): Constraint Satisfaction


YC1.1

EightQueenSolver
+ board: [8x8]
+ solve(): str + backtracking(col: int): boolean + isValid(row: int, col:int): boolean + printBoard(): str

- Khởi tạo bàn cờ 8x8 trống
- *solve()* sẽ gọi đệ quy backtracking(số cột sẽ tăng dần)
- *backtracking()* được đệ quy tối đa 8 lần để điền 8 quân hậu vào bàn cờ
- *isValid()* kiểm tra tính hợp lệ của quân hậu khi điền vào với trạng thái bàn cờ hiện tại
- *printBoard()* từ danh sách bàn cờ sẽ in ra cho người dùng xem

## Câu 1 (2.0 điểm): Constraint Satisfaction

YC1.2

 <b>NQueenSolver</b>
+ board: [NxN] + N: int
+ solve(): str + backtracking(col: int): boolean + isValid(row: int, col: int): boolean + printBoard(): str

- Khởi tạo bàn cờ với NxN ô trống
- *solve()* sẽ gọi đệ quy backtracking(số cột sẽ tăng dần)
- *backtracking()* được đệ quy tối đa 8 lần để điền 8 quân hậu vào bàn cờ
- *isValid()* kiểm tra tính hợp lệ của quân hậu khi điền vào với trạng thái bàn cờ hiện tại
- *printBoard()* từ danh sách bàn cờ sẽ in ra cho người dùng xem



## **Câu 2 – Adversarial Search**

### YC2.1

1. Mở với tên tệp đã cho để đọc.
2. Đọc dòng đầu tiên của tệp và gán giá trị cho `e` và `l` tương ứng.
3. Khởi tạo một từ điển trống có tên là `nodes_dict`.
4. Đối với `tôi` trong phạm vi (`e`), hãy làm như sau:
  - a. Đọc dòng tiếp theo của tệp và chia thành `a` và `b`.
  - b. Nếu `a` chưa phải là khóa trong `nodes_dict`, hãy thêm nó làm khóa mới với giá trị `Node(a)`.
  - c. Nếu `b` chưa phải là khóa trong `nodes_dict`, hãy thêm nó làm khóa mới với giá trị `Node(b)`.
  - d. Nối giá trị của `nodes_dict[b]` vào danh sách kế vị của `nodes_dict[a]`.
5. Đối với `tôi` trong phạm vi (`l`), hãy làm như sau:
  - a. Đọc dòng tiếp theo của tệp và chia nó thành nút và giá trị.
  - b. Đặt giá trị của `nodes_dict[node]` thành `int(value)`.
6. Đặt `self.root` thành `nodes_dict["n00"]`.
7. Đặt `self.terminalStates` thành một cách hiểu từ điển để lọc ra các nút có người kế vị và tạo một từ điển ánh xạ số nhận dạng của chúng với giá trị của chúng.
8. Đặt `self.successors` thành một cách hiểu từ điển để ánh xạ từng nút trong `nodes_dict` vào danh sách kế vị của nó.

YC2.2

```
def print_tree(self, node, level=0):  
    print("\t" * level, node)  
    for succ in node.successors:  
        self.print_tree(succ, level+1)
```

*Cài đặt hàm print()*

### YC2.3

#### *Minimax Algorithms*


```
function minimax(node, maximizingPlayer)
  if node is a terminal node
    return the heuristic value of node
  if maximizingPlayer
    bestValue :=  $-\infty$ 
    for each child of node
      v := minimax(child, FALSE)
      bestValue := max(bestValue, v)
    return bestValue
  else
    bestValue :=  $+\infty$ 
    for each child of node
      v := minimax(child, TRUE)
      bestValue := min(bestValue, v)
    return bestValue
```

```
function run()
  minimax(root, True)
  return result
```

*Cài đặt hàm run()*


## Câu 3 – Logical Agents

YC3.1

 <b>EightQueenSolver</b>
+ board: [8x8] + solver: Glucose3()
+ add_clause(clause) + solve()

*Cài đặt EightQueenSolver bằng Glucose3 với CNF*

YC3.2

 <b>NQueenSolver</b>
+ N: int + board: [NxN] + solver: Glucose3()
+ add_clause(clause) + solve()

*Cài đặt NQueenSolver bằng Glucose3 với CNF*

## **Câu 4 – Machine Learning**



YC4.1

### Decision Tree

**lib:** keras.datasets, sklearn.tree, sklearn.metrics, joblib, numpy

**preprocessing:**

+ method: reshape(data)

**input:** train data, test data

**train:**

+ initialize model: DecisionTreeClassifier()

+ fit model: fit (train data)

+ Predict model: predict(data)

**test:**

+ method: accuracy\_score(data)

**save model:** joblib.dump (model)

**load model:** joblib.load (model)

plot tree:

+ Create new figure: figure()

+ plot: plot\_tree()

YC4.2

### Naïve Bayes classifier

**lib:** keras.datasets, sklearn.naive\_bayes, sklearn.metrics, joblib, numpy.

**preprocessing:**

+ method: reshape(data)

**input:** train data, test data,

**train:**

+ initialize model: GaussianNB ()

+ fit model: fit (train data)

+ Predict model: predict(data)

**test:**

+ method: accuracy\_score(data)

**save model:** joblib.dump (model)

**load model:** joblib.load (model)

YC4.3

### KNeighborsClassifier

**lib:** keras.datasets, sklearn.neighbors, sklearn.metrics, joblib, numpy.

**preprocessing:**

+ method: reshape(data)

**input:** train data, test data, k

**train:**

+ initialize model: KNeighborsClassifier (k)

+ fit model: fit (train data)

+ Predict model: predict(data)

**test:**

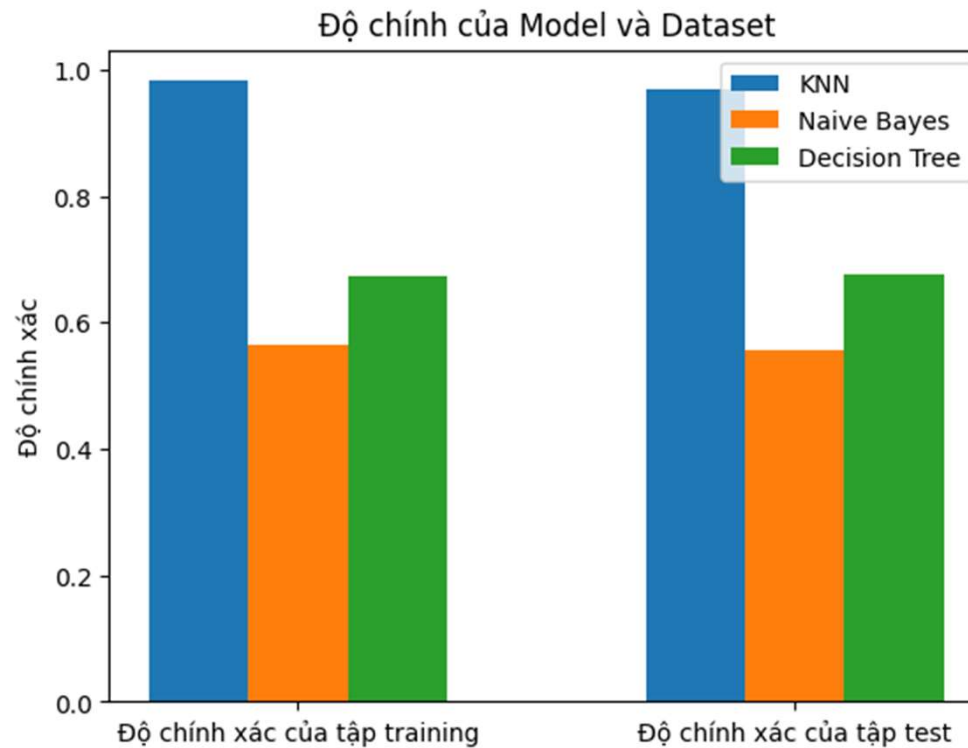
+ method: accuracy\_score(data)

**save model:** joblib.dump (model)

**load model:** joblib.load (model)

## Câu 4 (2.0 điểm): Machine Learning

YC4.4



*Thu thập các training và test accuracy ta vẽ được biểu đồ độ chính xác này bằng thư viện matplotlib.pyplot*

## Đánh giá tổng quát

Tiêu chí	Điểm	Đạt được
Câu 1	2 điểm	2 điểm
Câu 2	2 điểm	2 điểm
Câu 3	2 điểm	2 điểm
Câu 4	2 điểm	2 điểm
Câu 5	2 điểm	---
Tổng		---

## *Thuận lợi*

- Đa số các thành viên làm việc ăn ý
- Đa số các thành viên có kiến thức về Git
- Đa số thành viên nhiệt tình tham gia tìm hiểu đề tài
- Vấn đề nghiên cứu được phân chia bài bản
- Các thành viên có nền tảng lập trình tốt

## *Khó khăn*

- Số lượng thành viên nghiên cứu thực tế tương đối ít
- Mất nhiều thời gian để ôn tập lại python do các thành viên không xuất phát điểm từ các ngành khoa học máy tính
- Còn nhiều thiếu sót về phân tích đề
- Lượng công việc cho từng thành viên khá lớn
- Dù khó khăn nhưng nhóm vẫn hoàn thành toàn bộ bài tập

## References

### Book

- [1] *Stuart Russell, Peter Norvig. Artificial Intelligence : A Modern Approach, 2<sup>nd</sup> edition. New Jersey, McGraw-Hill, 2005*
- [2] Tom Mitchell. Machine Learning. New York, McGraw-Hill, 1997.
- [3] Manning and Schuetze, Foundations of Statistical Natural Language Processing, MIT Press. Cambridge, MA, 1999.

### Website

- [1] [sakai.it.tdt.edu.vn](http://sakai.it.tdt.edu.vn)