Ton Duc Thang University
Faculty of Information Technology

FINAL PROJECT

Course: Introduction to Artificial Intelligence

Duration: 06 weeks

I. Formation

- The project is conducted in groups of 03 – 05 students.
- Student groups conduct required tasks and submit the project following instructions below.

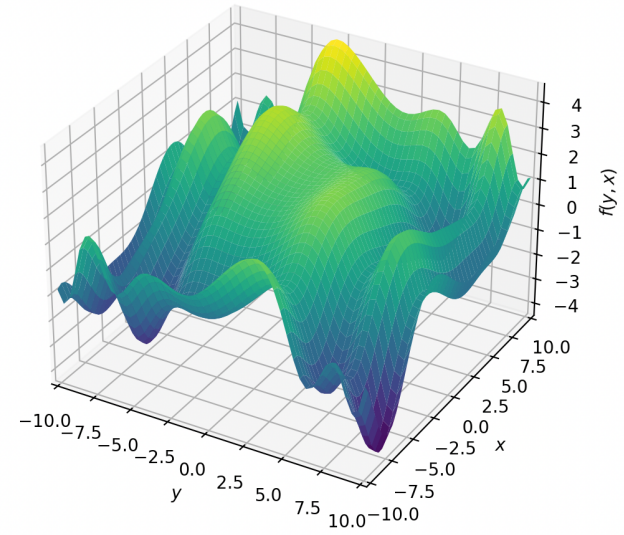
II. Tasks

a) Task 1 (2.0 point(s)): Simulated Annealing Search on 3D surfaces

Given a multivariable function

$$f(x,y) = \sin\left(\frac{x}{8}\right) + \cos\left(\frac{y}{4}\right) - \sin\left(\frac{x \cdot y}{16}\right) + \cos\left(\frac{x^2}{16}\right) + \sin\left(\frac{y^2}{8}\right)$$

- Students draw a 3D surface to illustrate the given function *f* using the `sympy` library, for instance,



- Then, implement the Simulated Annealing Search algorithm (SAS) to search for the location (x, y) with the f-value as large as possible.
  - Starting from O (0, 0),
  - Actions: for each state (x, y), the step size is 0 or $\pi/32$,

  - Students propose your own *schedule()* function to compute the *temperature T* given a time step *t*.
- Draw a red line connecting all visited points along the search path.
- Students organize the program regarding to the OOP model, ensure source code is compact and reasonable.
- Recommended editor: Visual Studio Code.

b) Task 2 (3.0 point(s)): 9x9 Tic-Tac-Toe with Heuristic Alpha-Beta Search

- Implement a program that allows users to play Tic-Tac-Toe against the computer on a 9 x 9 board, where whoever has 4 pieces in a row horizontally, vertically, or diagonally wins.
- The game operates on the console screen. The player selects a square by entering its coordinates from the keyboard. Students can update the board interface by erasing and redrawing it after each turn.
- The algorithm used for the computer is heuristic alpha-beta search.
  - *combine the original alpha-beta search algorithm and the h-minimax search,*
  - *students propose the depth limit L and the heuristic function estimating the h-minimax value, explicitly explain the rationality of the proposed heuristic function.*
- Students organize the program regarding to the OOP model, ensure source code is compact and reasonable.
- Recommended editor: Visual Studio Code.

c) Task 3 (3.0 point(s)): Constraint Satisfaction Problems with Propositional Logic

- Given a *m x n* matrix, each cell consists of a non-negative integer or it is blank.
- Each cell has 9 “adjacent” neighbors, including itself and 8 cells around.
- The player color cells by red or green colors so that the number of green cells which are “adjacent” to a cell matches the number inside.
- There is no constraint for blank cells.

Input data file (left) – Result (right)

- Students solve the given problem using propositional logic and the Glucose3 module of PySAT.
  - Assign a propositional symbol to each cell (true → green, false → red),
  - Enumerate cells to generate CNF clauses representing constraints,
  - Discover the general rule to generate clauses and eliminate redundant clauses,
  - Find a model satisfying all clauses using Glucose3 of PySAT,
- Implement a function to evaluate the result matrix and illustrate it on the console screen with colors.
- Students organize the program regarding to the OOP model, ensure source code is compact and reasonable.
- Recommended editor: Google Colab.

d) Task 4 (1.0 point(s)): Naïve Bayesian Classifier

- Given a data set of quiz scores and course results of students in *task4_data.csv*,
  - #: row index
  - Rank: P → pass, F → Fail
  - Q1, Q2, …, Q9: quiz scores
- Scores are continuous values. Therefore, students propose an approach to discretize these values and then manually implement the Naïve Bayesian classification algorithm.

- Finally, compute the accuracy of the classifier in the given data set.
- Students organize the program regarding to the OOP model, ensure source code is compact and reasonable.
- Recommended editor: Google Colab

e) Task 5 (1.0 point(s)): Report

- Student groups compose the project report using the IEEE conference proceeding template.
- Recommended editor: Overleaf.
- Selective contents:
  - *Title*: the project title
  - *Authors*: group member's information, the lecturer is appended as the last author.
  - *Abstract*: summarize the project requirements, approaches, experimental results, and levels of completion.
  - Each following section presents a task in the project, with a meaningful and human-readable title. Briefly introduce the approach to tackle the problem and illustrate results with related figures/tables, etc.
  - "*Contributions*" section: individual tasks, individual completion levels (0%-100%).
  - "*Self-evaluation*" section: self-evaluate task completion and estimate scores.
  - "*Conclusion*" section: summarize the project requirements, approaches, experimental results, and levels of completion.
- References are in the IEEE format.
- Maximal length is 05 pages.

III. Submission

- Create a folder whose name is in the format

final_<Group ID>_<Student ID>

  - **source:** source code files

- **report.pdf:** report of the project

- Students maintain outputs of all cells in ipynb files.
- Compress the folder into a zip file and submit by the deadline.
- Every team member must submit the project individually.

IV. Policy

- **Student groups submitting late get 0.0 points for each member.**
- **Copying source code on the internet/other students, sharing your work with other groups, etc. cause 0.0 points for all related groups.**
- **If there exist any signs of illegal copying or sharing of the assignment, then extra interviews are conducted to verify student groups' work.**
- **AI tools are forbidden in the project.**

-- THE END --