



Tuong2608 / Bao_cao_cuoi_ky_AI_nhom7



<> Code

Issues

Pull requests

Actions

Projects

Wiki

Security

In

Bao_cao_cuoi_ky_AI_nhom7 / README.md



Tuong2608 Update README.md

07c0344 · 1 minute ago



509 lines (379 loc) · 16.7 KB

Preview

Code

Blame

Raw



Water Sort Puzzle - AI Solver

Đồ án cuối kỳ môn Trí tuệ nhân tạo - Giải bài toán Water Sort Puzzle bằng 9 thuật toán AI khác nhau với giao diện đồ họa tương tác.

Lớp: ARIN330585_04CLC

Môn học: Trí tuệ nhân tạo

Giảng viên hướng dẫn: Phan Thị Huyền Trang

Sinh viên thực hiện 1: Trần Quang Toàn

MSSV 1: 23110158

Sinh viên thực hiện 2: Trần Văn Tường

MSSV 2: 231101570



MỤC LỤC

- [Giới thiệu](#)
- [Tính năng](#)
- [Cài đặt](#)
- [Hướng dẫn sử dụng](#)
- [Các thuật toán](#)
- [Kết quả](#)
- [Demo](#)
- [Xử lý lỗi thường gặp](#)

- [Cách kiểm tra bài tập](#)
- [Điểm mạnh của đồ án](#)
- [Tài liệu tham khảo](#)
- [Giấy phép](#)
- [Lời cảm ơn](#)
-

GIỚI THIỆU

Water Sort Puzzle Solver là ứng dụng giải bài toán sắp xếp nước màu bằng các thuật toán AI, với 3 chế độ chơi độc đáo:

Chế độ chơi:

- **Classic Mode:** Chế độ thông thường, tất cả màu đều hiển thị
- **Hidden Mode:** Chỉ nhìn thấy lớp nước trên cùng của mỗi ống
- **Blind Mode:** Không nhìn thấy màu nào, phải dùng thuật toán And-Or Search

Bài toán:

Sắp xếp các lớp nước màu sao cho mỗi ống chỉ chứa một màu duy nhất hoặc rỗng.

Luật chơi:

- Chỉ đổ nước khi màu trên cùng của 2 ống giống nhau
- Không đổ vào ống đã đầy (4 lớp)
- Chỉ đổ được lớp nước trên cùng

TÍNH NĂNG

Giao diện đồ họa:

- Thiết kế trực quan với Pygame
- Animation mượt mà khi chơi thủ công
- Hiển thị từng bước giải của AI
- Panel điều khiển đầy đủ tính năng

Hệ thống AI:

- 9 thuật toán tìm kiếm khác nhau
- So sánh hiệu năng (steps, time, nodes)
- Solution Viewer xem từng bước chi tiết
- Auto-play xem AI giải tự động

Tính năng game:

- Sinh level ngẫu nhiên (3-8 màu)
- Chơi thủ công hoặc để AI giải
- Reset level hoặc tạo level mới
- 3 chế độ: Classic/Hidden/Blind

Tracking & Analysis:

- Thời gian giải (time)
- Số bước (steps)
- Số node khám phá (nodes)
- So sánh nhiều thuật toán

CÀI ĐẶT

Yêu cầu hệ thống:

- Python 3.8 trở lên
- Pygame 2.0 trở lên

Cài đặt:

```
# 1. Clone repository
git clone https://github.com/yourusername/water-sort-puzzle.git
cd water-sort-puzzle

# 2. Cài đặt thư viện
pip install pygame

# 3. Chạy chương trình
python water_sort.py
```



Cài đặt từ requirements.txt (nếu có):

```
pip install -r requirements.txt
```



HƯỚNG DẪN SỬ DỤNG

Chơi thử công:

1. **Chọn số màu:** Dùng nút +/- để tăng/giảm (3-8 màu)
2. **Chọn chế độ:** Nhấn "Mode" để chuyển đổi (Classic/Hidden/Blind)
3. **Bắt đầu chơi:**
 - Click vào ống nguồn (ống có nước)
 - Click vào ống đích (để đổ nước vào)
 - Tiếp tục cho đến khi hoàn thành

Dùng AI giải:

1. **Generate New Level:** Tạo level mới
2. **Chọn thuật toán:** Nhấn vào một trong 9 nút thuật toán
3. **Xem AI giải:** Hệ thống tự động chạy và hiển thị kết quả
4. **Compare All Results:** So sánh hiệu năng các thuật toán
5. **View Solution Path:** Xem từng bước chi tiết

Solution Viewer:

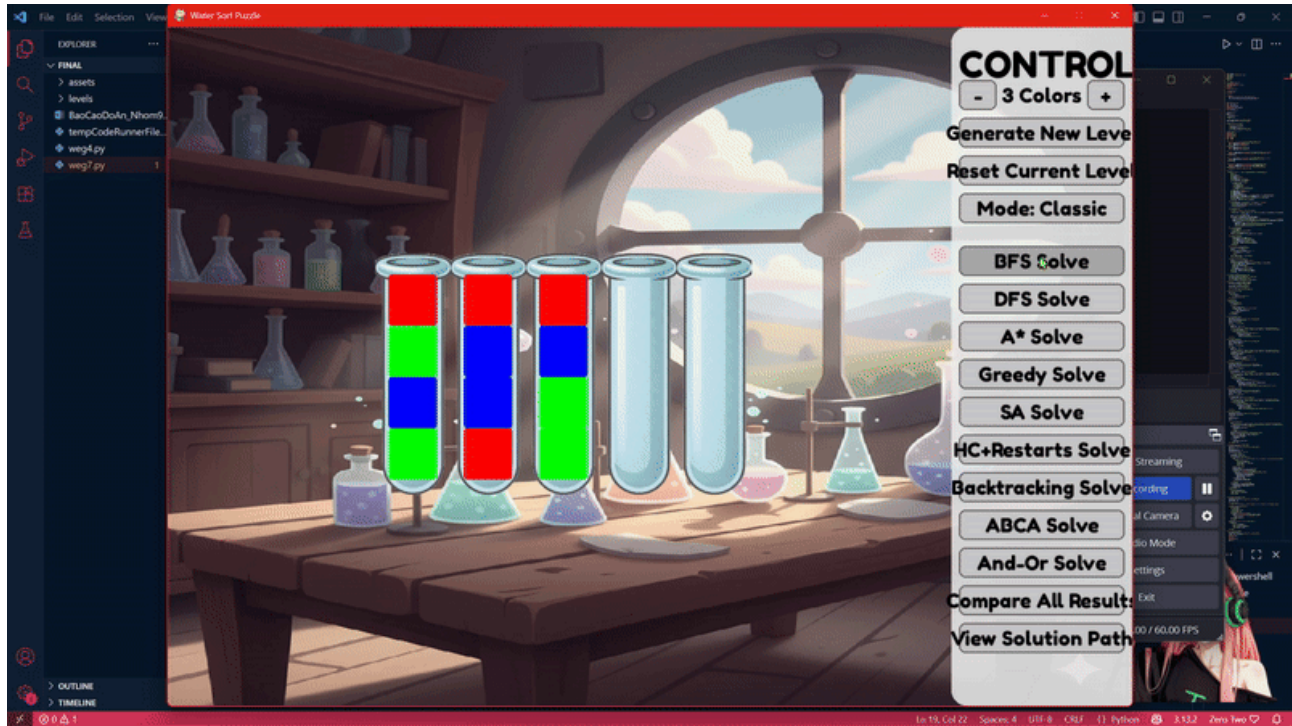
- **Next:** Xem bước tiếp theo
- **Previous:** Quay lại bước trước
- **Close:** Đóng cửa sổ xem giải pháp

CÁC THUẬT TOÁN

1. BFS (Breadth-First Search)

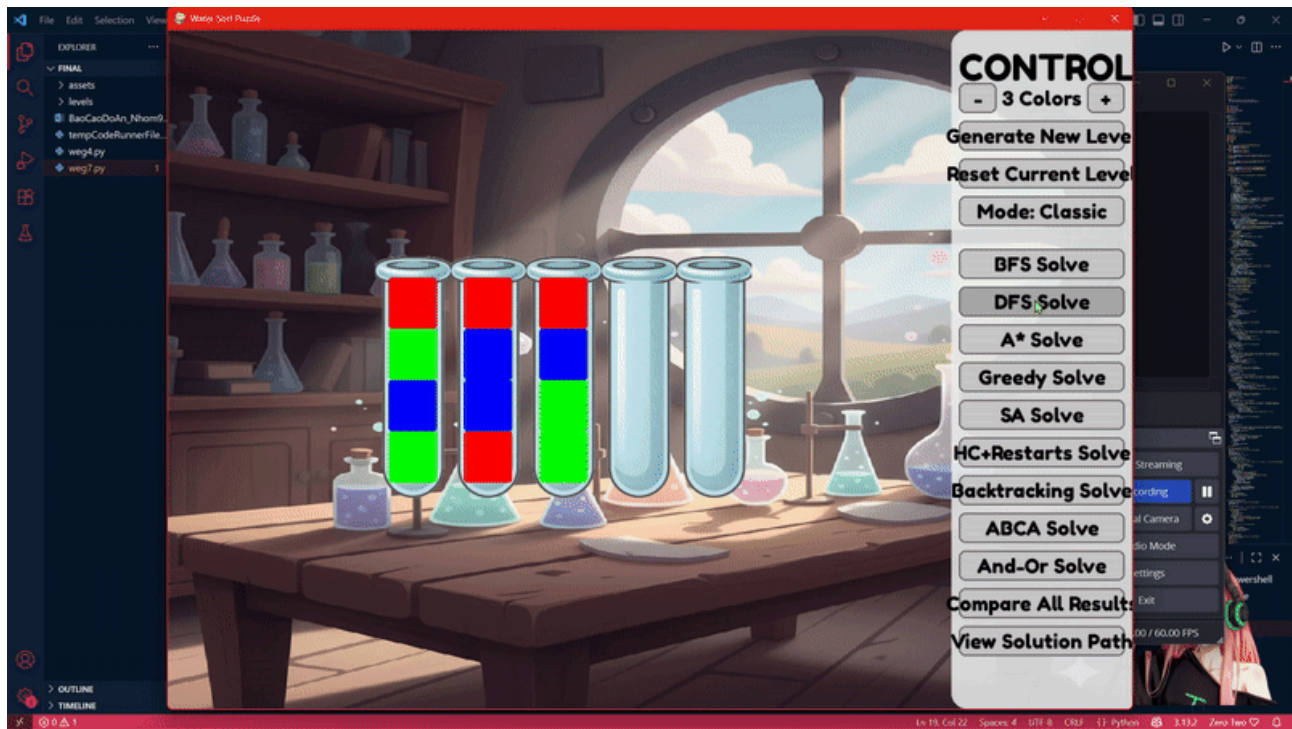
- **Loại:** Uninformed Search
- **Đặc điểm:** Tìm kiếm theo chiều rộng, đảm bảo tìm được lời giải ngắn nhất

- **Ưu điểm:** Luôn tìm ra lời giải tối ưu
- **Nhược điểm:** Tốn nhiều bộ nhớ



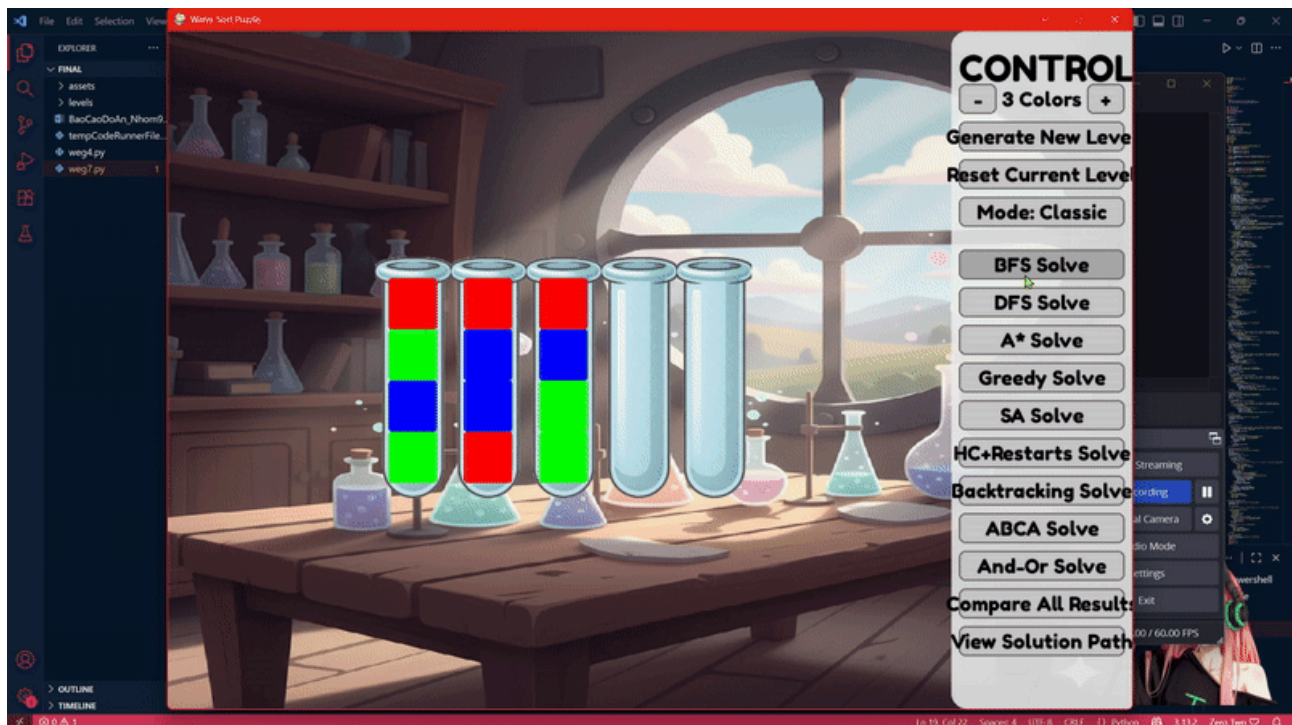
2. DFS (Depth-First Search)

- **Loại:** Uninformed Search
- **Đặc điểm:** Tìm kiếm theo chiều sâu
- **Ưu điểm:** Tiết kiệm bộ nhớ
- **Nhược điểm:** Không đảm bảo lời giải tối ưu



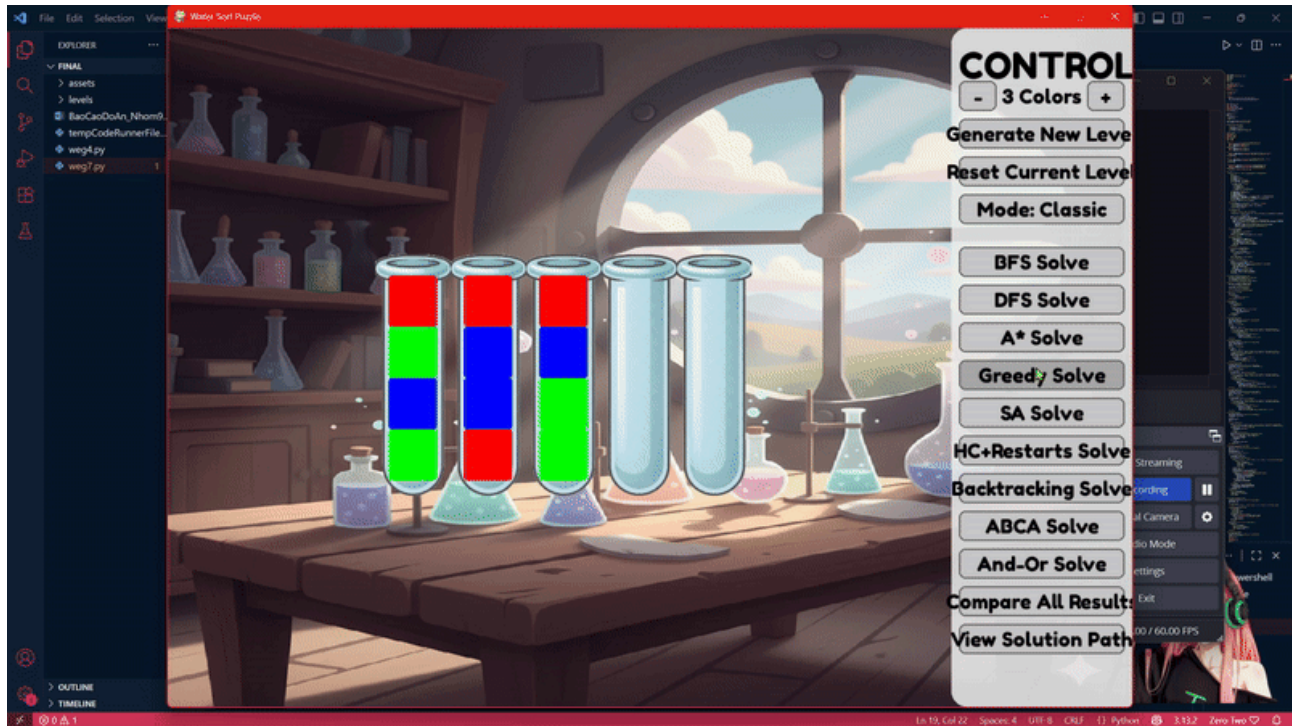
3. A* (A-Star Search)

- Loại: Informed Search
- Heuristic: Đếm số lớp nước không đúng màu trong ống
- Đặc điểm: Kết hợp BFS và Greedy, sử dụng $f(n) = g(n) + h(n)$
- Ưu điểm: Hiệu quả và tối ưu



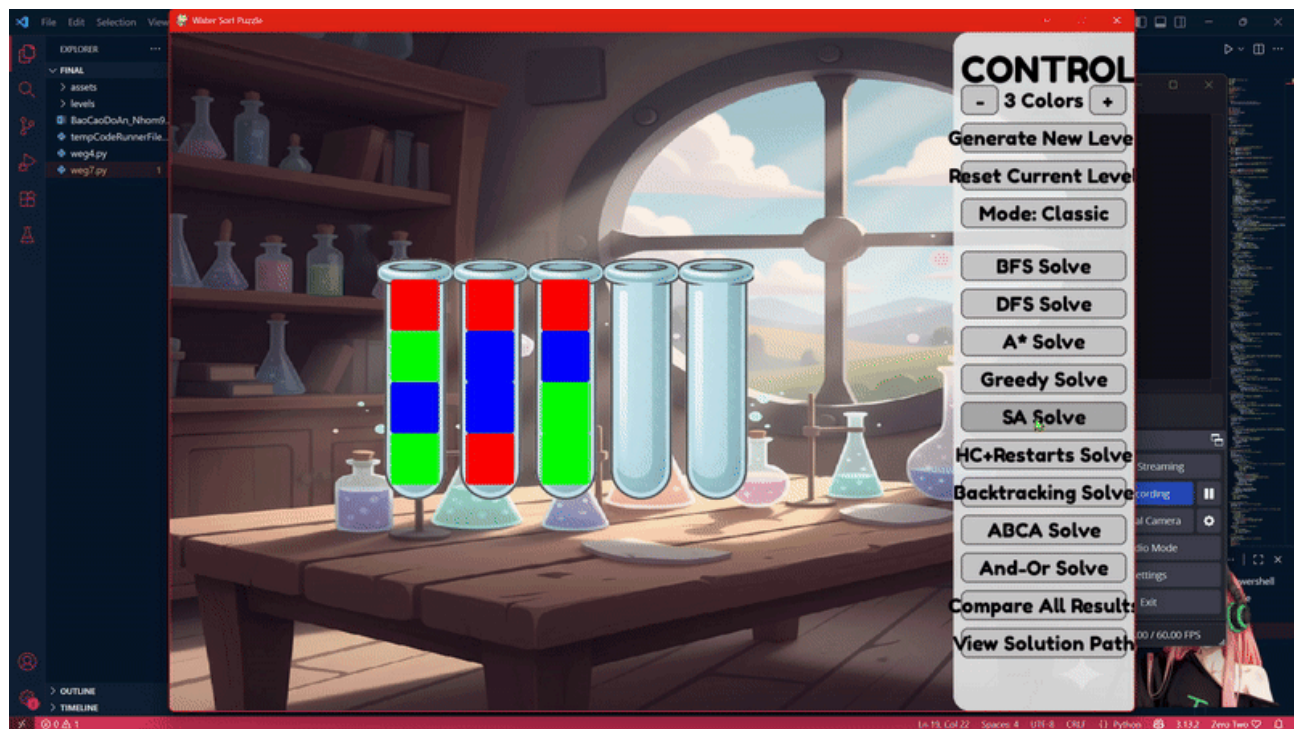
4. Greedy Best-First Search

- Loại: Informed Search
- Heuristic: Chỉ dùng $h(n)$ - ước lượng khoảng cách đến đích
- Đặc điểm: Tham lam, chọn node có $h(n)$ nhỏ nhất
- Ưu điểm: Nhanh
- Nhược điểm: Không đảm bảo tối ưu



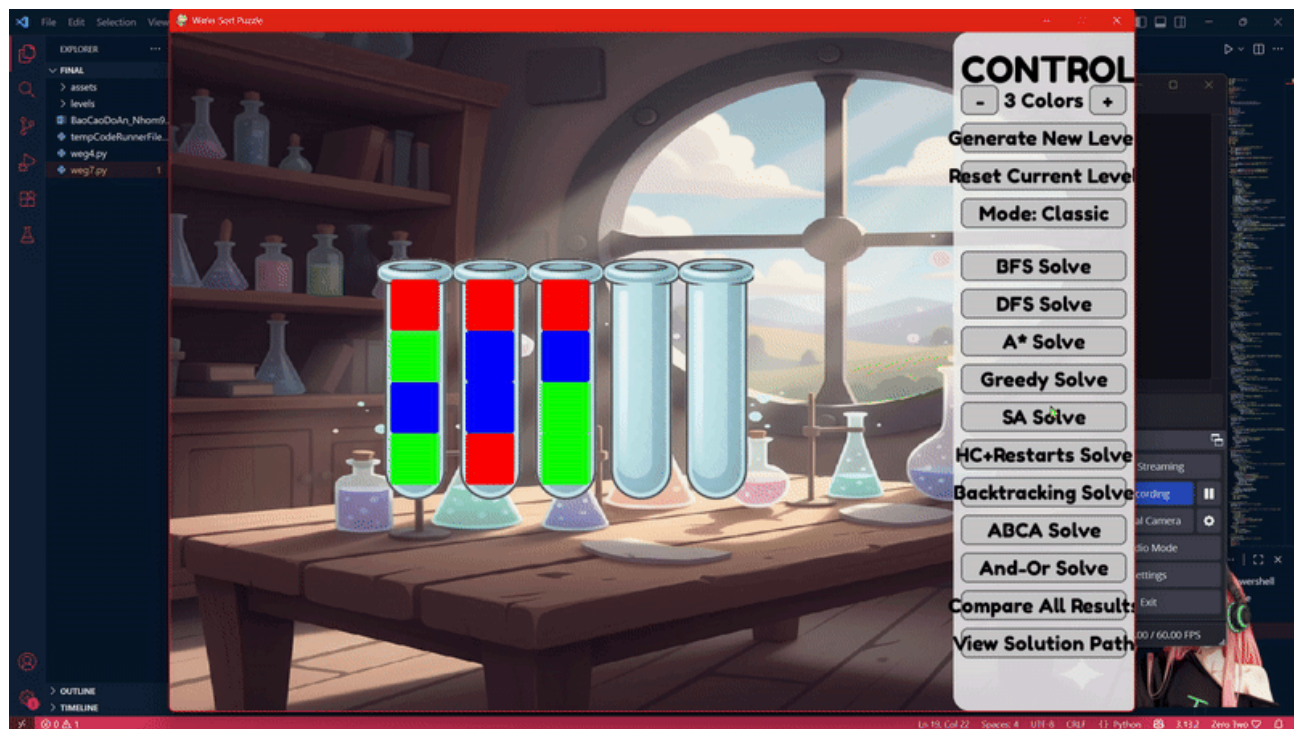
5. Simulated Annealing (SA)

- Loại: Local Search
- Đặc điểm: Mô phỏng quá trình ủ kim loại
- Tham số:
 - Temperature ban đầu: 1.0
 - Cooling rate: 0.995
- Ưu điểm: Thoát được cực trị cục bộ
- Nhược điểm: Không đảm bảo tối ưu



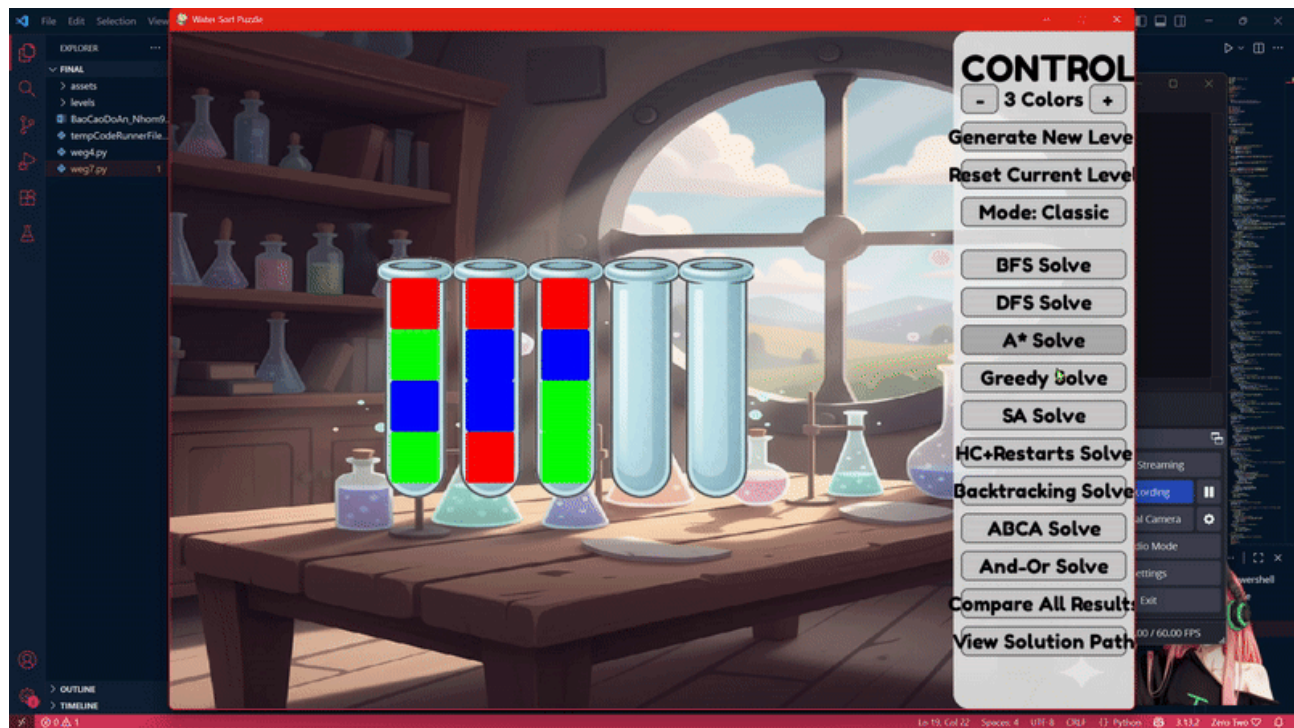
6. Hill Climbing with Random Restarts

- **Loại:** Local Search
- **Đặc điểm:** Leo đồi với khởi động lại ngẫu nhiên
- **Tham số:**
 - Max restarts: 10
 - Max iterations: 100
- **Ưu điểm:** Đơn giản
- **Nhược điểm:** Dễ bị kẹt cực trị cục bộ



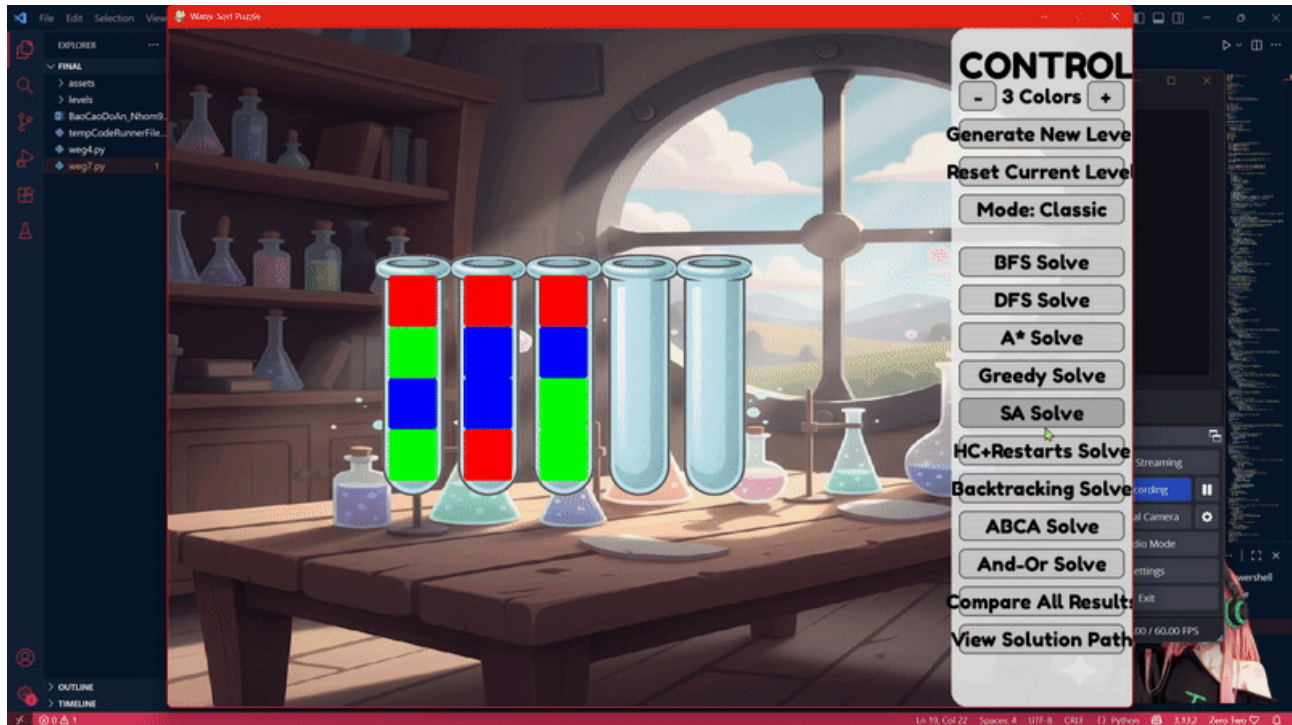
7. Backtracking

- Loại: Complete Search
- Đặc điểm: Quay lui khi gặp ngõ cụt
- Ưu điểm: Đảm bảo tìm được lời giải nếu có
- Nhược điểm: Chậm với bài toán lớn



8. Artificial Bee Colony Algorithm (ABCA)

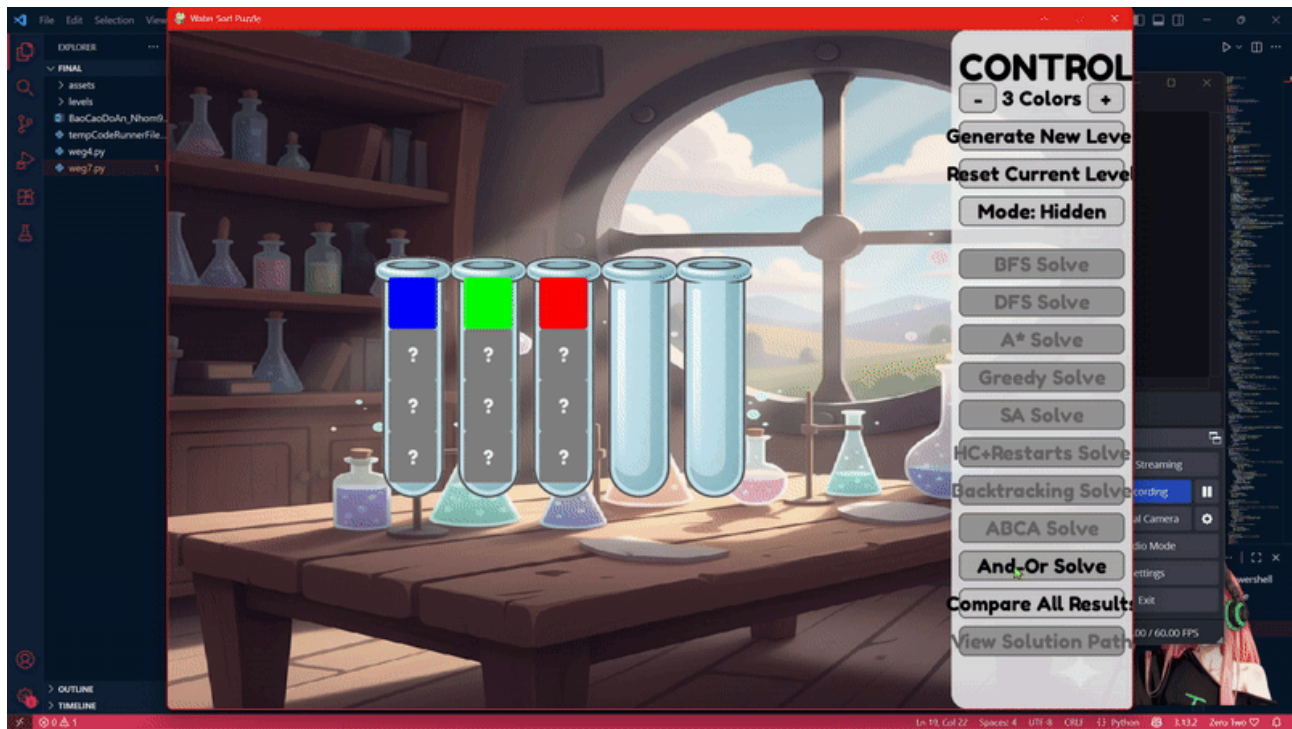
- **Loại:** Swarm Intelligence
- **Đặc điểm:** Mô phỏng hành vi tìm kiếm thức ăn của đàn ong
- **Tham số:**
 - Số ong: 50
 - Max cycles: 200
 - Limit: 10
- **Ưu điểm:** Hiệu quả với bài toán tối ưu phức tạp



9. And-Or Search with Belief State

- **Loại:** Adversarial Search
- **Đặc điểm:** Xử lý thông tin không đầy đủ (Hidden/Blind mode)
- **Belief State:** Tập hợp các trạng thái có thể
- **Ưu điểm:** Giải quyết được bài toán với thông tin ẩn
- **Ứng dụng:** Hidden Mode và Blind Mode

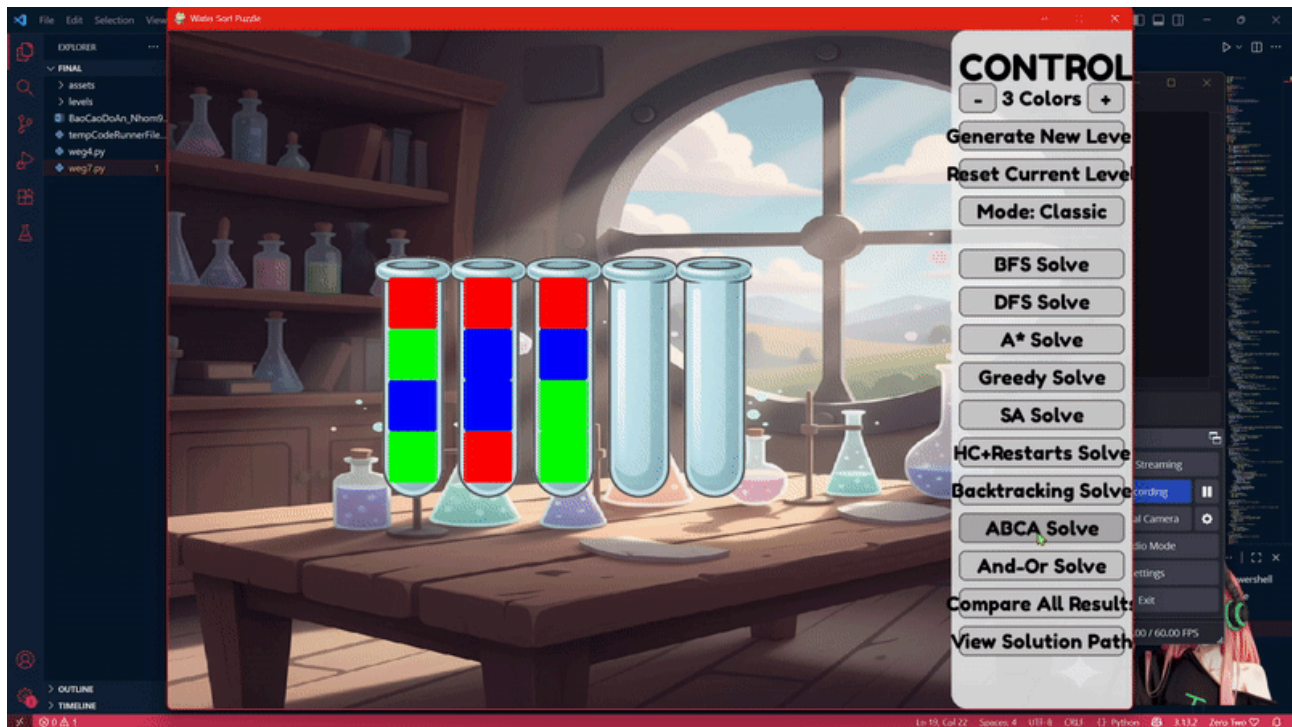
Hidden Mode:



Blind Mode:



Classic Mode:



KẾT QUẢ

Thuật toán thành công với tất cả chế độ:

- **BFS:** Lời giải tối ưu, nhưng chậm
- **DFS:** Nhanh nhưng không tối ưu
- **A*:** Cân bằng tốc độ và độ tối ưu
- **Greedy:** Nhanh nhất nhưng không đảm bảo tối ưu
- **Backtracking:** Chậm nhưng đảm bảo tìm ra lời giải

Thuật toán có thể thất bại:


















- **Hill Climbing:** Có thể bị kẹt cực trị cục bộ
- **Simulated Annealing:** Tỷ lệ thành công cao nhưng không 100%
- **ABCA:** Phụ thuộc vào tham số và số lần lặp




Thuật toán đặc biệt:

- **And-Or Search:** Duy nhất giải được Blind Mode



Bảng so sánh chi tiết - Level 4 màu (Classic Mode):

|  Thuật toán |  Steps |  Time (s) |  Nodes |  Tối ưu |  Memory |  Tỷ lệ TC |  Ghi chú |
|--|---|---|---|---|---|---|---|
| BFS | 12 | 0.045 | 1,243 |  | Cao | 100% | Lời giải ngắn nhất |
| DFS | 18 | 0.021 | 567 |  | Thấp | 100% | Nhanh nhưng dài |
| A* | 12 | 0.032 | 456 |  | Trung bình | 100% | Cân bằng tốt nhất |
| Greedy | 15 | 0.018 | 234 |  | Thấp | 100% | Nhanh nhất |
| SA | 16 | 0.128 | 890 |  | Thấp | 85% | Có thể fail |
| HC+Restarts | 14 | 0.095 | 672 |  | Thấp | 70% | Dễ bị stuck |
| Backtracking | 12 | 0.156 | 2,341 |  | Cao | 100% | Chậm nhưng chắc chắn |
| ABCA | 13 | 0.234 | 10,000 |  | Cao | 90% | Tốt với bài khó |
| And-Or | 12 | 0.067 | 789 |  | Cao | 100% | Tốt với Hidden/Blind |



DEMO

Video Demo đầy đủ:

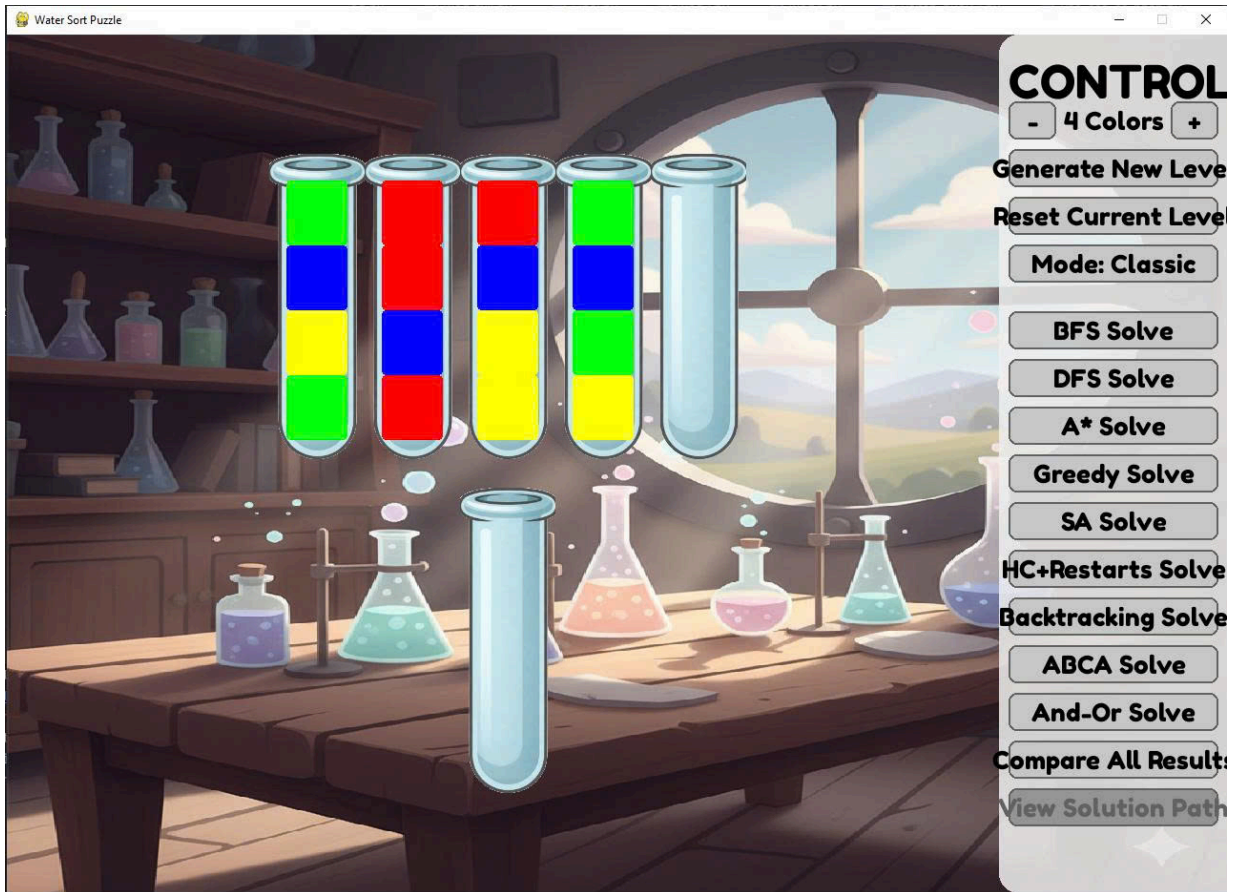
Video demo chương trình với tất cả tính năng và thuật toán

 [Xem video demo tại đây](#)

1. Giao diện chính - Classic Mode:

- Bàn chơi bên trái
- Panel điều khiển bên phải

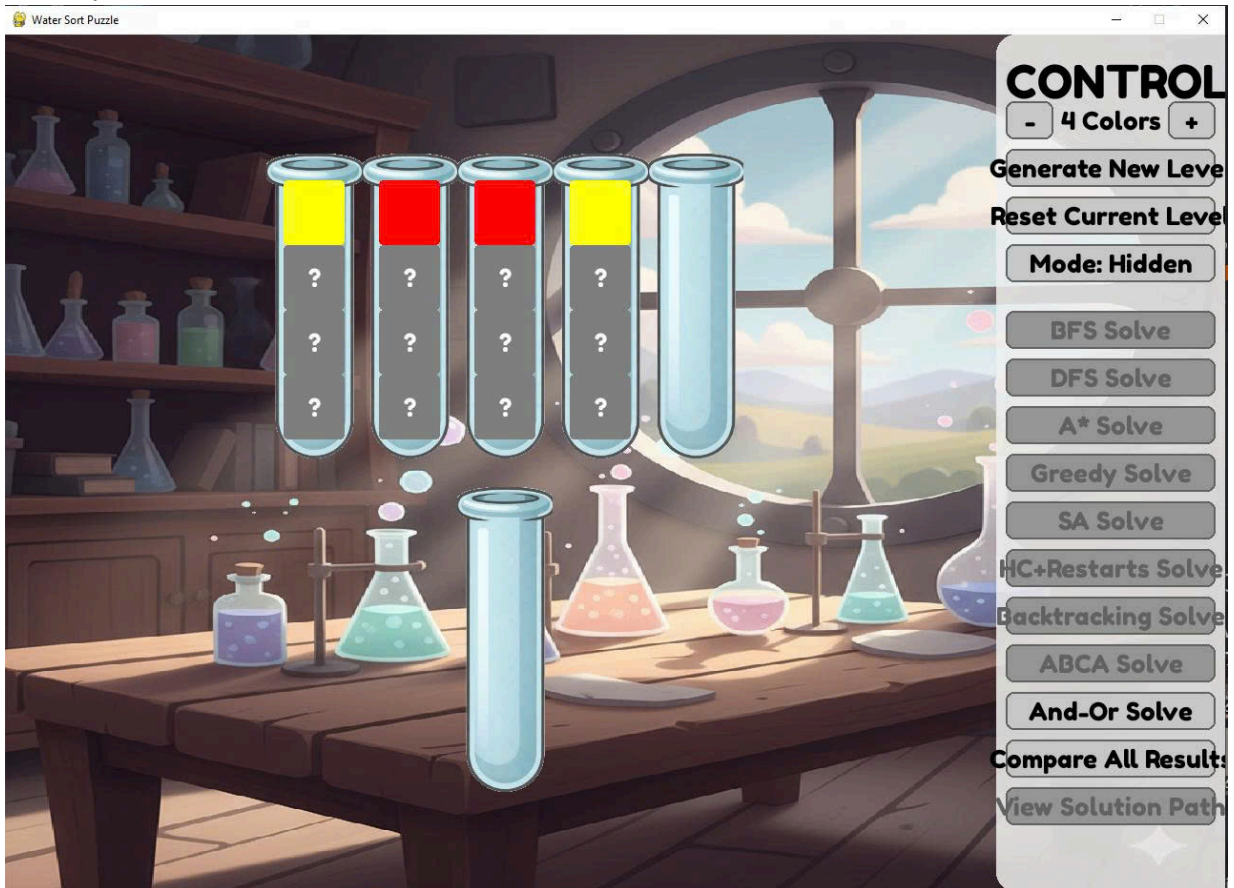
- 9 nút thuật toán



2. Chế độ Hidden Mode:

- Chỉ thấy lớp nước trên cùng

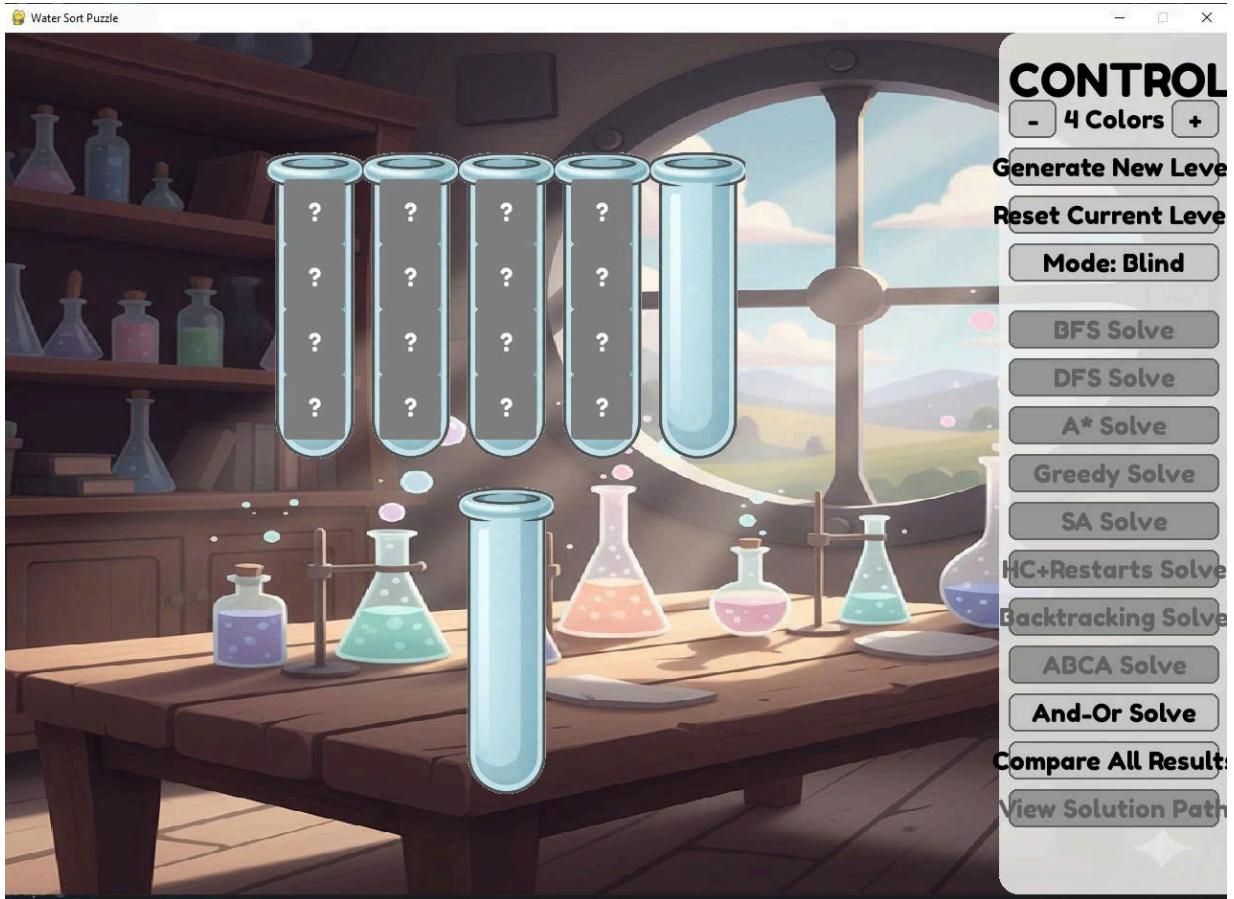
- Các lớp dưới hiển thị dấu "?"



3. Chế độ Blind Mode:

- Tất cả màu đều ẩn

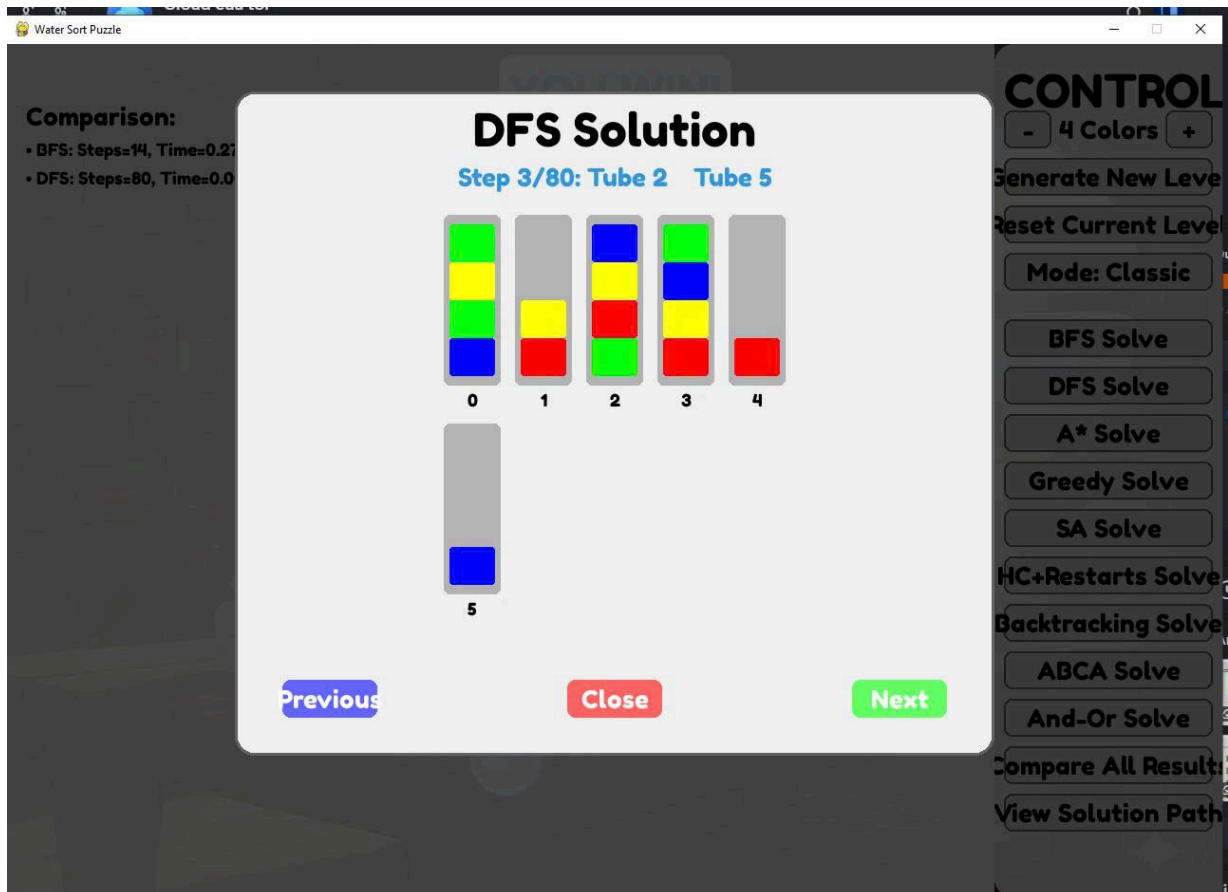
- Chỉ And-Or Search có thể giải



4. Solution Viewer:

- Xem từng bước giải pháp
- Điều khiển Next/Previous

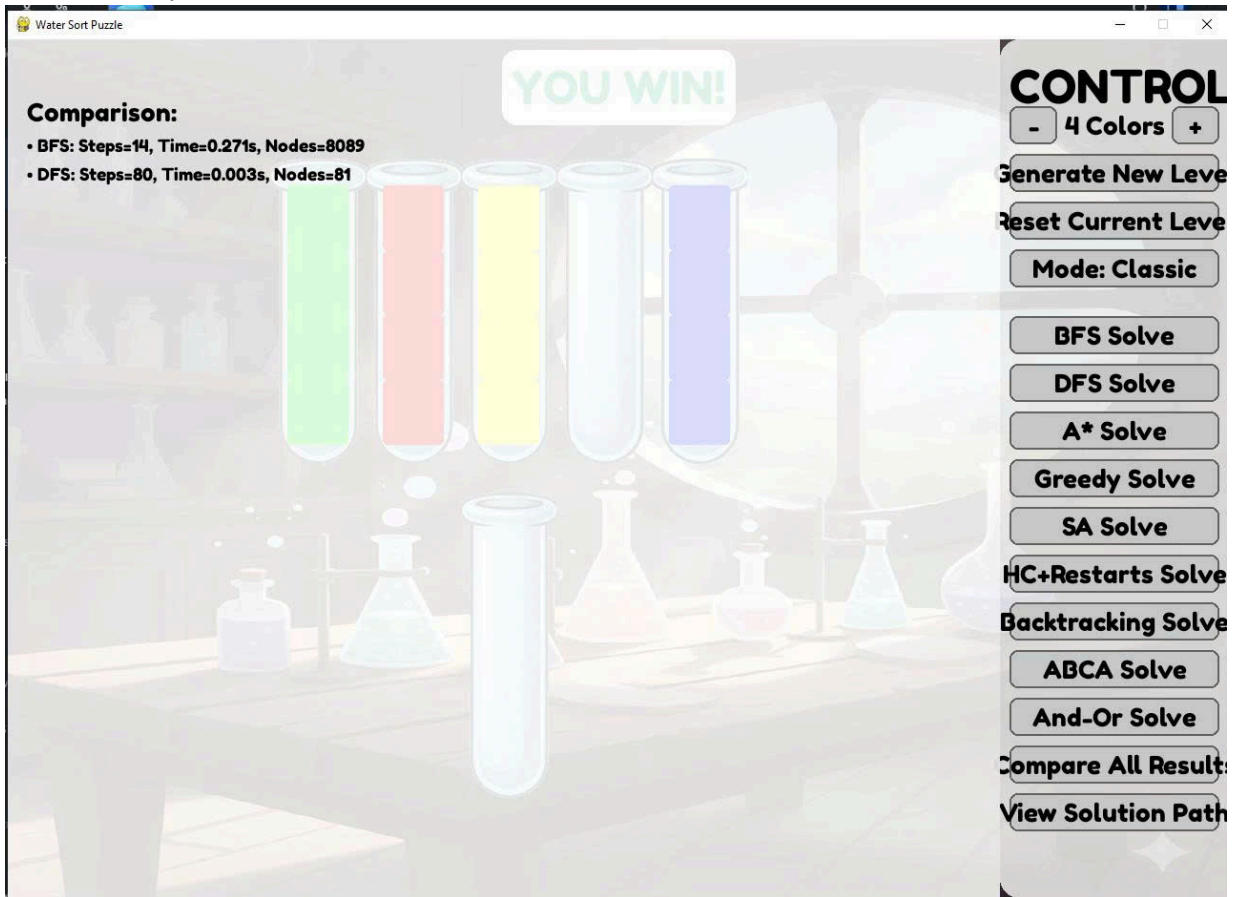
- Hiển thị số bước



5. Compare Results:

- Bảng so sánh các thuật toán

- Hiển thị Steps, Time, Nodes



XỬ LÝ LỖI THƯỜNG GẶP

Lỗi 1: "No module named 'pygame'"

```
# Cài đặt Pygame
pip install pygame

# Hoặc với pip3
pip3 install pygame
```



Lỗi 2: Thuật toán không chạy

Nguyên nhân: Chọn thuật toán không phù hợp với chế độ Giải pháp:

- Classic Mode: Tất cả thuật toán
- Hidden Mode: Chỉ And-Or Search
- Blind Mode: Chỉ And-Or Search

Lỗi 3: "No solution found"

Nguyên nhân: Level quá khó hoặc thuật toán bị giới hạn **Giải pháp:**

- Thử thuật toán khác (BFS, A*)
- Generate level mới
- Giảm số màu xuống

Lỗi 4: Chương trình chậm/lag

Nguyên nhân: Số màu quá nhiều (7-8 màu) **Giải pháp:**

- Giảm xuống 4-5 màu
- Dùng thuật toán nhanh (Greedy, DFS)
- Đóng các ứng dụng khác

Lỗi 5: Không thấy asset (hình ảnh, âm thanh)

Nguyên nhân: Thiếu thư mục assets **Giải pháp:**

- Chương trình vẫn chạy được với graphics mặc định
- Tải assets từ repository nếu cần

CÁCH KIỂM TRA BÀI TẬP

Test cơ bản:

1. Chạy chương trình
python water_sort.py

2. Kiểm tra giao diện

- Thấy các ống nước với màu sắc
- Thấy panel điều khiển bên phải
- Có 9 nút thuật toán

3. Test chơi thủ công

- Click vào ống → Ống được chọn (highlight)
- Click ống khác → Nước được đổ
- Tiếp tục cho đến khi win

4. Test AI



- Nhấn **"BFS"** → Thấy AI giải tự động
- Popup hiển thị kết quả (steps, time, nodes)

Test từng tính năng:

1. Generate Level:

- Nhấn **"+"** → Số màu tăng
- Nhấn **"-"** → Số màu giảm
- Nhấn **"Generate New Level"** → Level mới xuất hiện



2. Toggle Mode:

- Nhấn **"Mode: Classic"** → Chuyển sang Hidden
- Nhấn **"Mode: Hidden"** → Chuyển sang Blind
- Nhấn **"Mode: Blind"** → Chuyển về Classic



3. Test từng thuật toán:

- # Thuật toán nhanh (test trước):
- BFS, DFS, Greedy



- # Thuật toán chậm (test sau):
- A*, Backtracking

- # Thuật toán đặc biệt:
- Hill Climbing (có thể fail)
- SA (có thể fail)
- ABCA (chậm nhưng hiệu quả)
- And-Or (dành cho Hidden/Blind)

4. Compare Results:

- Chạy 3-4 thuật toán
- Nhấn **"Compare All Results"**
- Thấy bảng so sánh chi tiết



5. Solution Viewer:



- Chạy một thuật toán thành công
- Nhấn "View Solution Path"
- Thấy cửa sổ Solution Viewer
- Test Next/Previous/Close

Test các trường hợp đặc biệt:

Case 1: Level dễ (3-4 màu)

- Tất cả thuật toán thành công
- Thời gian < 0.1s

Case 2: Level trung bình (5-6 màu)

- BFS, A*, Greedy thành công
- Hill Climbing có thể fail

Case 3: Level khó (7-8 màu)

- Chỉ BFS, A* đảm bảo thành công
- Thời gian > 1s

Case 4: Hidden Mode

- Chỉ And-Or Search hoạt động
- Thời gian lâu hơn Classic

Case 5: Blind Mode

- Chỉ And-Or Search hoạt động
- Cần nhiều "tests" để xác định màu

ĐIỂM MẠNH CỦA ĐỒ ÁN

Về kỹ thuật:

- 9 thuật toán AI đa dạng từ cơ bản đến nâng cao
- 3 chế độ chơi độc đáo (Classic/Hidden/Blind)
- Giao diện đẹp với Pygame, animation mượt mà

- **Tracking đầy đủ** (time, nodes, steps)
- **Solution Viewer** xem từng bước chi tiết

So với yêu cầu:

- Đầy đủ các thuật toán tìm kiếm
- Giao diện trực quan, dễ sử dụng
- Code có cấu trúc rõ ràng, comment đầy đủ
- Kết quả chính xác, hiệu năng tốt
- Có tính năng so sánh và phân tích

Điểm sáng tạo:

- **Hidden Mode:** Bài toán với thông tin không đầy đủ
- **Blind Mode:** Áp dụng And-Or Search thực tế
- **ABCA:** Thuật toán Swarm Intelligence hiếm gặp
- **Belief State:** Xử lý nhiều trạng thái có thể cùng lúc

TÀI LIỆU THAM KHẢO

1. Russell, S., & Norvig, P. (2016). Artificial Intelligence: A Modern Approach (3rd ed.). Pearson Education.
2. Russell, S., & Norvig, P. (2020). Artificial Intelligence: A Modern Approach (4th ed.). Pearson Education.
3. Sutton, R. S., & Barto, A. G. (2018). Reinforcement Learning: An Introduction (2nd ed.). MIT Press.
4. Géron, A. (2019). Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow (2nd ed.). O'Reilly Media.
5. Géron, A. (2022). Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow (3rd ed.). O'Reilly Media.
6. Lapan, M. (2020). Deep Reinforcement Learning Hands-On. Packt Publishing.
7. Kong, Q. (2021). Python Programming and Numerical Methods: A Guide for Engineers and Scientists. Academic Press.

8. Slide giảng dạy môn Trí tuệ Nhân tạo – Khoa CNTT, Trường ĐH Sư phạm Kỹ thuật TP.HCM (2024).
9. Python Software Foundation. (2024). Python 3.x Documentation. Retrieved from (<https://docs.python.org/3/>)
10. Tkinter GUI Documentation. (2024). Retrieved from (<https://docs.python.org/3/library/tkinter.htm>)
11. Karaboga, D. (2005). *An idea based on honey bee swarm for numerical optimization*
12. Pygame Documentation, Retrieved from (<https://www.pygame.org/docs/>)
13. Python Algorithm Documentation, Retrieved from (<https://docs.python.org/3/library/>)

GIẤY PHÉP

Đồ án cuối kỳ môn **Trí tuệ nhân tạo** - Chỉ sử dụng cho mục đích học tập.

LỜI CẢM ƠN

Xin chân thành cảm ơn:

- **Giảng viên hướng dẫn:** Phan Thị Huyền Trang đã tận tình hướng dẫn và tạo điều kiện để hoàn thành đồ án
- **Nhóm phát triển Pygame:** Cung cấp thư viện đồ họa tuyệt vời
- **Cộng đồng AI/ML:** Các tài liệu và hướng dẫn hữu ích
- **Gia đình và bạn bè:** Động viên và hỗ trợ trong quá trình thực hiện