

Project part 3

Refactor 1

Refactoring 1: Extract Method (isIsolatedNode)

Type: Extract Method

What: Moved the isolated node logic from saveToDotFile into a new method isIsolatedNode.

Why: Improves readability, reusability, and maintainability of the saveToDotFile method.

Link: https://github.com/abdullahmasood6/CSE464_2024_ahakim1/compare/main...refactor

Refactor 2

Refactoring 2: Simplify Logic (constructPath)

Type: Simplify Logic

What: Refactored the constructPath method to use an ArrayList and Collections.reverse for better readability and efficiency.

Why: Improved clarity and performance by avoiding the use of LinkedList.addFirst. Link:

https://github.com/abdullahmasood6/CSE464_2024_ahakim1/commit/8361fc2eb6ce43688bfc0676327b242135624272

Refactor 3

Refactoring 3: Extract Method (validateNodeInput)

Type: Extract Method

What: Simplified the addEdge method by moving node validation logic into a reusable method validateNodeInput.

Why: Improved readability and reusability while ensuring consistent validation across the codebase.

https://github.com/abdullahmasood6/CSE464_2024_ahakim1/commit/6931d3ef7358b9cb56172bf86545a4b17e2c880a

Refactor 4

Refactoring 4: Extract Method (processDotFileLine)

Type: Extract Method

What: Moved the logic for parsing .dot file lines from loadFromDotFile into a helper method processDotFileLine.

Why: Improved readability by separating file reading and line processing logic, making both parts more modular and maintainable.

Link: https://github.com/abdullahmasood6/CSE464_2024_ahakim1/commit/461c58d4f76d6cfd001566fb3e0f8585d240228d

Refactor 5

Refactoring 5

Refactoring 5: Extract Method (removeEdgesForNode)

Type: Extract Method

What: Moved the logic for removing edges connected to a node from `removeNode` into a helper method removeEdgesForNode.

Why: Improved readability by simplifying the `removeNode` method and made the edge removal logic reusable for future needs.

Commit Link:

https://github.com/abdullahmasood6/CSE464_2024_ahakim1/commit/09a8db09c3ae89379aec7e82a0693da7237233f6

Part 2

Type: Apply Design Pattern (Template Method)

What: Refactored the BFS and DFS implementations using the Template Method Pattern.

Common traversal logic (e.g., initializing the search, processing nodes, and constructing paths) was moved to an abstract base class (GraphSearchTemplate). BFS-specific and DFS-specific logic (e.g., traversal order) were implemented in concrete subclasses (BFSGraphSearch and DFSGraphSearch).

Why:

- Eliminated duplicate code by centralizing shared logic in the base class.
- Improved maintainability by separating common and algorithm-specific behaviors.
- Simplified adding future traversal algorithms (e.g., Random Walk) through extensibility of the base class.

• **Commit Link:**

https://github.com/abdullahmasood6/CSE464_2024_ahakim1/commit/34643abeb9c5c3e2baf5dc5fde9a913c021d862d

Part 3

Refactoring 7: Strategy Pattern for BFS and DFS

- Type: Apply Design Pattern (Strategy Pattern)

- What: Refactored the graphSearch method to use the Strategy Pattern. Created a GraphSearchStrategy interface and two concrete implementations: BFSGraphSearchStrategy and DFSGraphSearchStrategy.

Why:

Enabled dynamic selection of BFS or DFS at runtime.

Improved code organization by isolating BFS and DFS logic in separate strategy classes.

Enhanced flexibility for adding future algorithms (e.g., A*, Dijkstra).

https://github.com/abdullahmasood6/CSE464_2024_ahakim1/commit/9961d525ce7c0e676eccb692af76937e480209b4

Part 4

Refactoring 8: Random Walk Search Algorithm

Type: New Feature and Design Pattern Implementation

What: Added a new random walk search algorithm that selects the next node to visit randomly from the neighbors of the current node. Implemented using the Strategy Pattern.

Why:

Demonstrates the flexibility of the Strategy Pattern by adding a new algorithm without modifying existing logic. Allows for random exploration of graph paths to find the destination node.

How:

Created a RandomWalkGraphSearchStrategy class implementing the GraphSearchStrategy interface.

Updated the graphSearch method to support Algorithm.RANDOM_WALK.

Pull request to merge refactor branch to main for project part3 #15

abdullahmasood6 wants to merge 9 commits into main from refactor

Conversation 0 Commits 9 Checks 1 Files changed 11 +340 -65

abdullahmasood6 commented 19 hours ago

No description provided.

hakimabdullahmasood added 9 commits yesterday

- Refactor 1 e515a7d
- Refactor 2 8361fc2
- Refactor 3 6931d3e
- Refactor 4 461c58d
- Refactor 5 09a8db0
- Applied Template Method Pattern for BFS and DFS refactoring. 34643ab
- Implemented Strategy Pattern for BFS and DFS. 9961d52
- Added Random Walk Search algorithm using Strategy Pattern. 2fb1f5d
- Small update ✓ 0c199e3

Reviewers

No reviews

Still in progress? Learn about draft PRs

Assignees

No one—assign yourself

Labels

None yet

Projects

None yet

Milestone

No milestone

Development

Successfully merging this pull request may close these issues.

None yet

Notifications

Customize

Unsubscribe

You're receiving notifications because you're watching this repository.

1 participant

Lock conversation

Add a comment

Write Preview

Markdown is supported Paste, drop, or click to add files

Close pull request Comment