LangGraph: State and Reducers

State in LangGraph

Definition of State

State in LangGraph represents the *shared memory* that flows through the workflow. It acts as the central data container that holds all the information being passed and updated between nodes as the graph executes.

Conceptual Overview

- The **state** evolves as different nodes perform operations and produce new results.
- Each node may modify one or more parts of the shared state.
- This design enables **data persistence, reusability, and synchronization** across the workflow.

Example State Structure

Example: Essay Evaluation State

```
essay_text: str
topic: str
depth_score: int
language_score: int
clarity_score: int
total_score: int
feedback: Annotated[list[str], add]
evaluation_round: int
```

Explanation:

- `essay_text` and `topic` store the core textual inputs.
- `depth_score`, `language_score`, and `clarity_score` hold individual metric values.
- `total_score` is typically computed as a weighted sum or aggregate of the above.
- `feedback` contains a list of comments, where each node can append new insights.
- `evaluation_round` tracks the number of evaluation cycles.

Key Insight

LangGraph's state serves as a single source of truth throughout the workflow, ensuring that all nodes operate on the latest and most consistent information.

Reducers in LangGraph

Definition of Reducers

Reducers determine how updates from individual nodes are applied to the shared state. Each key in the state can have a corresponding reducer that controls whether the new data **replaces**, **merges**, or **appends** to the existing value.

Reducer Behavior Examples

- **Replace Reducer:** Overwrites the previous value with the latest node output.
- **Merge Reducer:** Combines structured data (e.g., dictionaries or JSON) without removing existing keys.
- **Add Reducer:** Appends new items (e.g., list entries such as feedback or logs) to the existing collection.

Illustration

```
# Example Reducer Behavior

"feedback": add          # new feedback is appended
"total_score": replace   # total score is updated
"metadata": merge        # combines previous and new data
```

Practical Insight

Reducers empower developers to precisely control how information evolves within the workflow — enabling dynamic state transitions, history tracking, and modular workflow design.

Summary

- The **State** represents a dynamic and shared data structure that travels through the workflow.
- **Reducers** define how updates are integrated, ensuring controlled and predictable data evolution.

- Together, they form the backbone of LangGraph's data management system, facilitating intelligent, adaptive, and traceable workflows.

"State holds the memory; reducers shape its evolution."