Documentation

Overview

This version of the code constructs a hierarchical graph based on a custom ranking system—in this case, the U.S. Air Force rank hierarchy. Each node in the graph corresponds to a rank, represented not by its name, but by a symbolic representation (e.g., stars for generals, different symbols for enlisted ranks).

The logic is similar to previous versions that used letters:

- Each character in the input string (`A`, `B`, `C`, …, up to `V`) maps to a specific rank level.
- The hierarchy is top-down, with `A` representing the highest rank (General of the Air Force) and subsequent letters representing progressively lower ranks.
- Sibling nodes (nodes of the same level under the same parent) are connected horizontally to show their relationship.
- The `#` character breaks the sibling chain, ensuring that a node following `#` will not link to its previous sibling.

Key Concepts

1. **Hierarchy by Rank:**
   - The code uses a predefined list of U.S. Air Force ranks.
   - Each letter (`A` through `V`) corresponds to one of the 22 ranks, from highest to lowest.
2. **Parent-Child Relationships:**
   - A node representing a particular rank connects to the most recently created node of the immediately higher rank.
   - For instance, if a node corresponds to “Lieutenant General (O-9)” (C-level), it connects to the most recently placed “General (O-10)” (B-level) node.
3. **Sibling Connections:**
   - When multiple nodes share the same parent and appear consecutively, they link to each other as siblings.

   - This sibling link is a horizontal connection in the hierarchy, providing visual continuity and clearly grouping children of the same parent.

4. **Breaking Sibling Chains with `#`:**
   - The `#` character acts as a reset switch for sibling connections.
   - After encountering `#`, the next rank node will connect to its parent but not to its previous sibling, effectively starting a new sibling chain from that point onward.

5. **Symbolic Representation:**
   - Instead of displaying the full rank names, each node is labeled with a symbolic representation.
   - For example, a 5-star rank might be shown as "★★★★★", while enlisted ranks are shown using different symbols.
   - This allows for a more compact and visually distinctive representation of the hierarchy.

Data Structures

- `rank_symbols`: A list of symbols corresponding to each rank level. The index in this list matches the rank level derived from the character.
- `last_node_of_letter`: Maps each rank level to the last created node of that level, allowing the code to determine the correct parent node for new nodes.
- `parent_last_child`: Maps a parent node to its most recently created child, enabling sibling links between consecutive children of the same parent.

Process Flow

1. **Input String Parsing:**
   - The input string is examined character by character.
   - Letters (`A` to `V`) are converted to a rank level index and thus to a rank symbol.
   - Nodes are created and linked to their parents (if not top-level).
   - Sibling links are created between consecutive siblings unless a `#` break occurs.

2. **`#` Reset Mechanism:**

   - On encountering `#`, sibling linking information is cleared.
   - The subsequent node after `#` connects only to its parent, not its previous sibling.
   - After placing that node, sibling linking resumes normally for subsequent nodes.

3. **Graph Rendering:**

   - The code uses `pygraphviz` for hierarchical layout (top-to-bottom) and `matplotlib` for visualization.
   - Each node displays a rank symbol.
   - Edges are drawn without arrowheads, focusing on hierarchy and sibling structure rather than direction.

4. **Checking Order with `check_string_allowed`:**

   - Before building the graph, you can run `check_string_allowed(input_string)` to ensure the string introduces ranks in proper alphabetical order of first appearance.
   - Once all letters are introduced in the correct order, they can reappear in any sequence.

Example with Visualization

- Consider an input `"ABBCABB"`:

  - `A` might represent the top rank (General of the Air Force).
  - `B` corresponds to the next rank (General, O-10).
  - Additional `B`s connect under `A` and link to each other as siblings.
  - Later, encountering `C` would connect to the last `B`.
  - Another `A` introduces a new top-level node, and subsequent `B`s under that `A` form another sibling chain.

- Adding a `#` (e.g., `"AB#B"`) breaks the sibling chain, so the second `B` after `#` does not form a sibling link with the previous `B`.

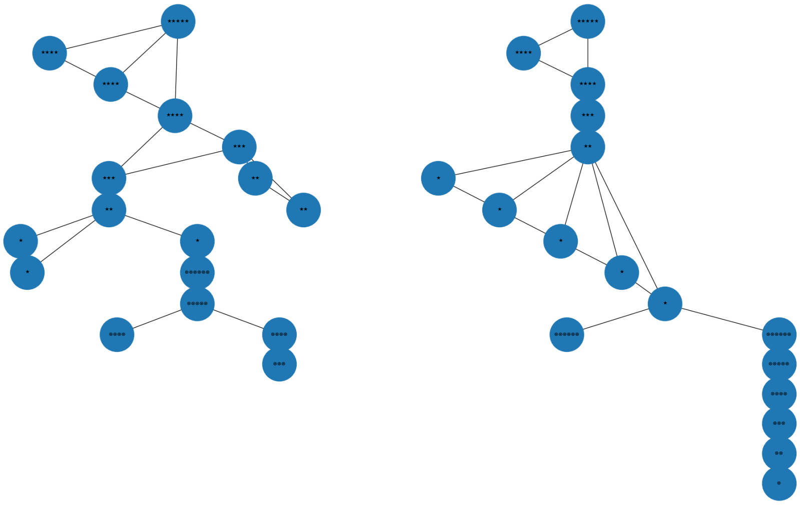

Figure 1: Example of ABBBCDDCDEE#EFGH#HIABBCDEEEEEF#FGH#I#J#K

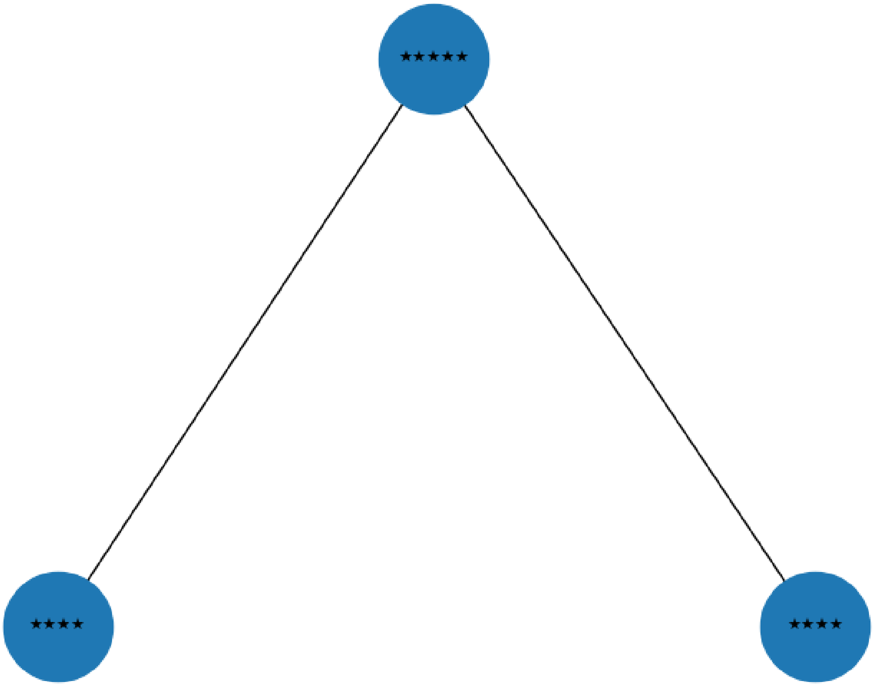

Figure 2: Example of AB#B