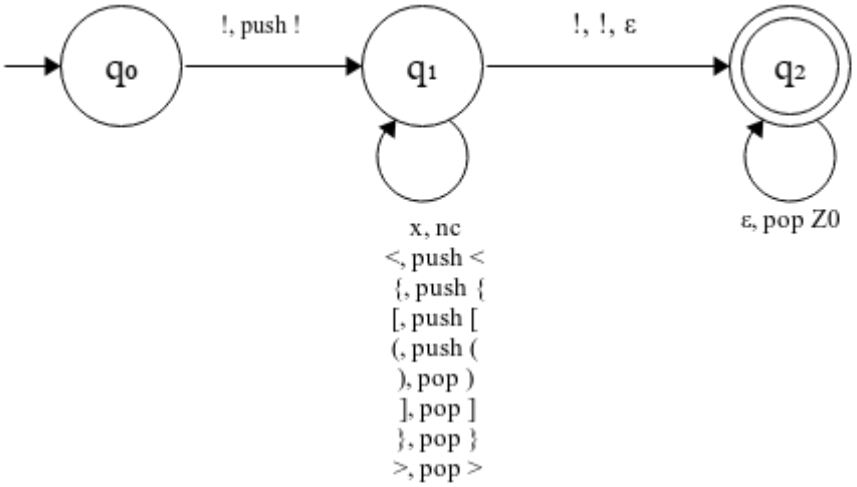
Evangalista, Abdiel
Lim, Hannah Marie

Project: PDA Multiple Balanced Brackets



The **PDA consists of three states: q0 (initial), q1 and q2 (final)**. Transitions are defined as follows:

State q0: Pushes ! to the stack upon reading !, transitioning to q1.
State q1: Processes input symbols and manipulates the stack:

- Pushes opening brackets (<, {, [, () onto the stack.
- Pops the stack when encountering closing brackets (>, },],)) if the top of the stack matches.
- On reading ! again, pops ! and transitions to q2.

State q2: Accepts the input if the stack is empty except for the bottom marker Z0.

The **Stack Alphabet Γ consists of: { Z0, !, <, {, [, (}** where Z0 is the initial bottom marker, ! marks the boundaries of the bracketed expression, and where <, {, [, (are pushed when their respective opening brackets are encountered in the input.

This PDA is implemented in python through a series of functions that simulate state transitions, stack operations, and input processing. The most important logic is in the *is_balanced function*, which iteratively processes the input string while managing the PDA's state and stack. The state variable tracks the current state **('q0', 'q1', or 'q2')**, and the stack which is in list form is initialized with the bottom marker **'Z0'** to know when the stack is empty.

Transitions are governed by checks in a loop that consumes the input string character by character. In **q0**, the PDA expects the opening **!** and when it is encountered, **!** is pushed to the stack, and the state transitions to **q1**. In **q1**, opening brackets **(<, {, [, ()** are pushed to the stack, while closing brackets trigger a pop only if the top of the stack matches the expected symbol. Mismatches return the *pos_error_message*, which highlights the failure position and remaining input. The *process_message* helper function logs the current instantaneous description (ID) of the PDA, displaying the state, unprocessed input, and stack contents. When a closing **!** is encountered in **q1**, the PDA transitions to **q2** if the stack's top matches **!**. The final check in **q2** ensures the stack contains only 'Z0' and nothing else. If valid, *success_message* confirms acceptance; otherwise, *state_error_message* returns a flag.

The *evaluate* function in the PDA parses valid expressions between the **!** delimiters, and also counts the number of x's. This function also uses a secondary stack (xStack) to compute the number of x's based on the following rules: > doubles the count, } increments by 1,] discards the count, and) subtracts 1 (minimum 0). This logic mirrors a PDA's stack behavior.

The *main1* and *main2* functions handle the batch processing of input lines. *main1* runs *is_balanced* on each line, printing detailed traces via *process_message*, while *main2* suppresses trace output using contextlib.redirect_stdout and only prints evaluation results for valid inputs.