HALTING User's Manual

About HALTING

Welcome to the HALTING project (Honors Automata Language Theory Including Navigable Graphics). This is an application designed and developed by Maya Montgomery for her undergraduate senior honors thesis in Computer Science at Hamilton College. This program simulates finite automata for the purpose of computer science language theory education; users can build and run FAs with the convenience of a graphical interface.

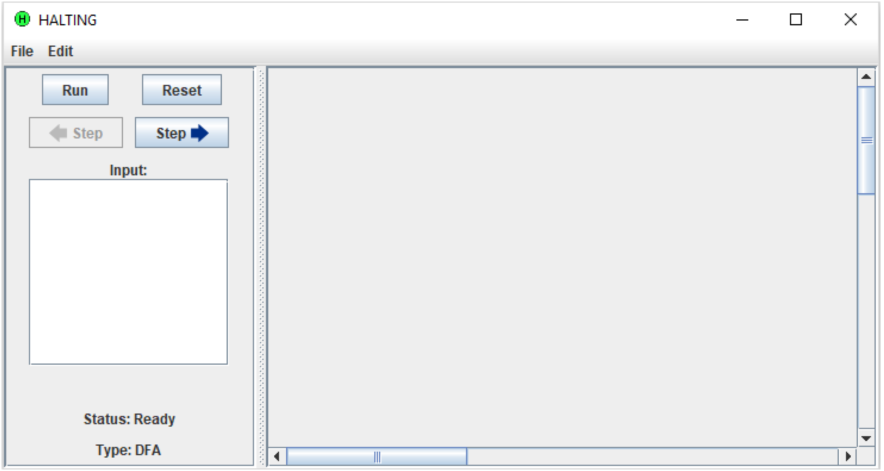
How to Use HALTING

Running the application:

HALTING is packaged as a runnable JAR file named HALTING.jar. Just double-click the file to run. You may need to grant executable permissions; you should be prompted with a pop-up window on Windows or Mac OS, whereas on Linux you may need to set this in a terminal (try `chmod a+x HALTING.jar`). Alternatively, you can enter `java -jar HALTING.jar` in a terminal to run the application. Please note that Java 1.8 or higher is required to run this program.

Building an automaton:

To start a new automaton, run the application to open a blank window, or select File > New to open another blank window.

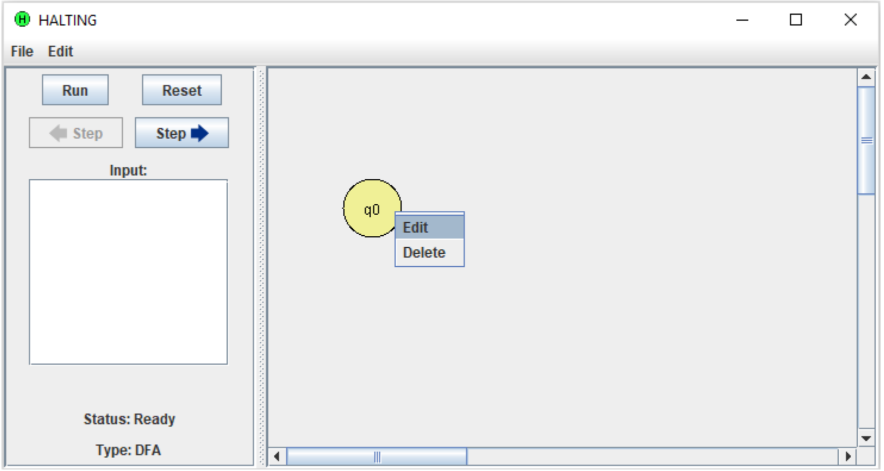


Status: Ready

Type: DFA

Add a new state by clicking on an empty spot on the canvas, which is on the right-hand side of the window. You may move states around on the canvas by clicking and dragging them (on a touchpad, double tap and drag), or by clicking them and then clicking an empty spot on the canvas. Click a selected state again to deselect it. Selected states are indicated by color.

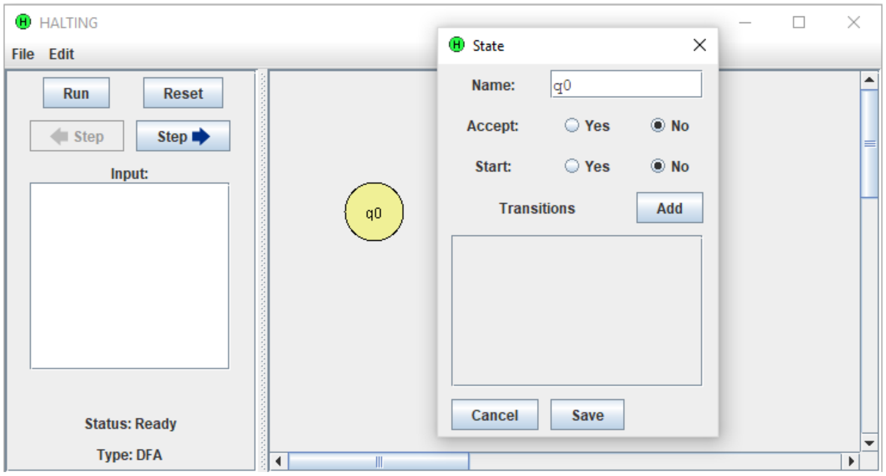
States can be edited or deleted by right-clicking them (or holding down the Ctrl key and clicking them) and selecting Edit or Delete.



Status: Ready

Type: DFA

When editing a state, a window opens with options to change the state's name, accept state status (true or false), start state status (true or false - note that an automaton can only have one start state), and transitions. From this window, select Save to close the window and apply any changes to the state, or choose Cancel to discard any changes.



Name: q0

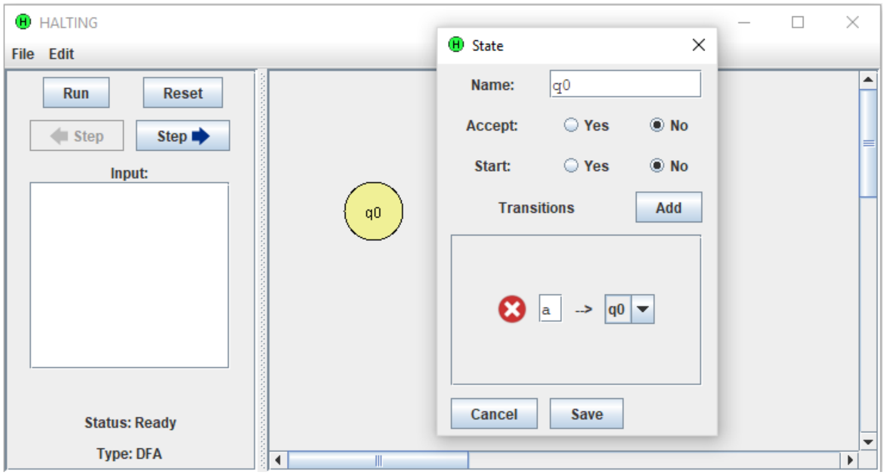
Accept: ○ Yes ● No

Start: ○ Yes ● No

Status: Ready

Type: DFA

Add a new transition by clicking ‘Add’ to the right of ‘Transitions’ in the state window, then enter the input character in the text field and select a target state from the dropdown menu. Delete a transition by clicking the X on the left of the transition.



Name: q0

Accept: ○ Yes ● No

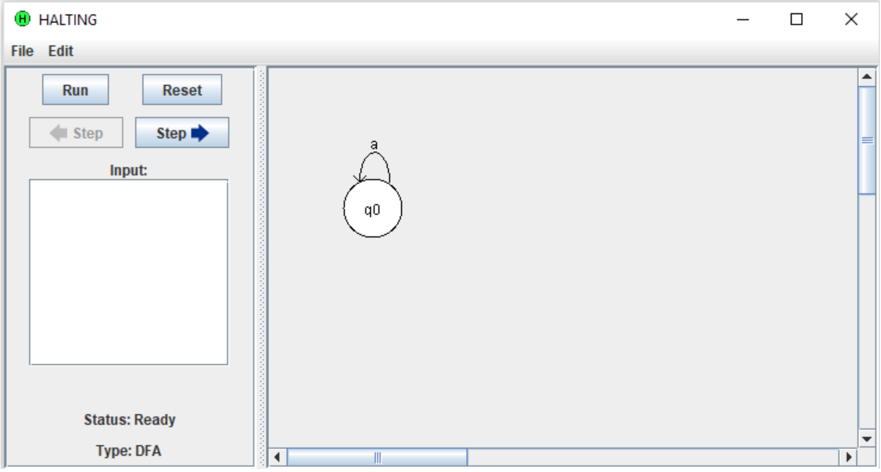
Start: ○ Yes ● No

a --> q0

Status: Ready

Type: DFA

State transitions are marked by arrows pointing from a state to its target state, with the transition’s input character(s) indicated at the arrow midpoints. A state that transitions to itself will have a circular arrow directly above it.

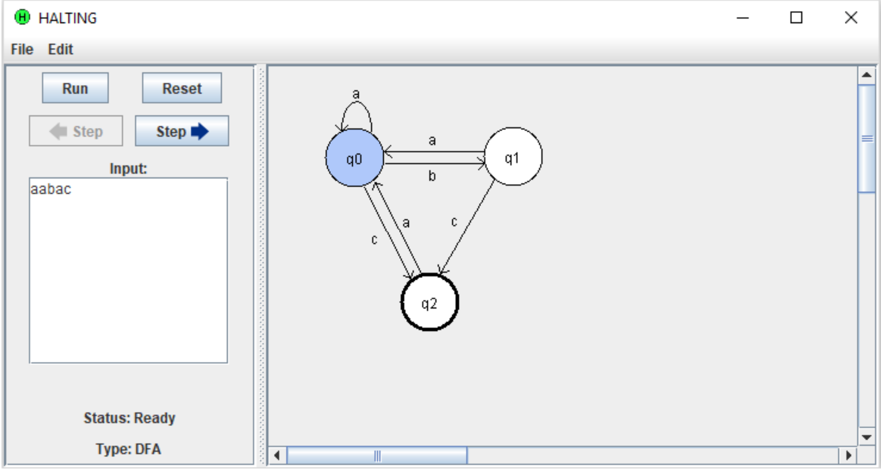


Status: Ready

Type: DFA

Running an automaton:

Enter the input string for the automaton in the text box located in the left-hand sidebar. Once the machine is running, this input cannot be edited until after the machine is reset using the Reset button. Note that a newline is considered a character (i.e. do not enter a newline after the input string).

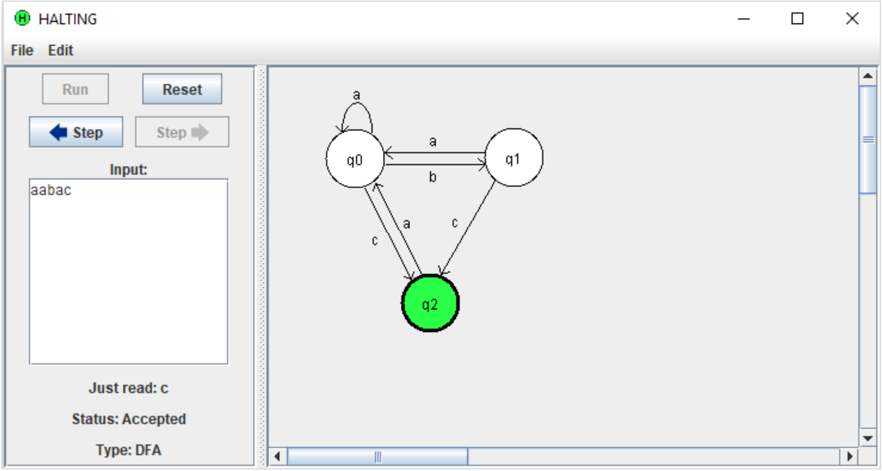


Input:
aabac

Status: Ready

Type: DFA

Option 1: Select Run to run the entire input through the automaton; either an error will be reported (e.g. if there is no transition defined from the current state on the current input character) or the final state will be indicated along with the accept status of the input.



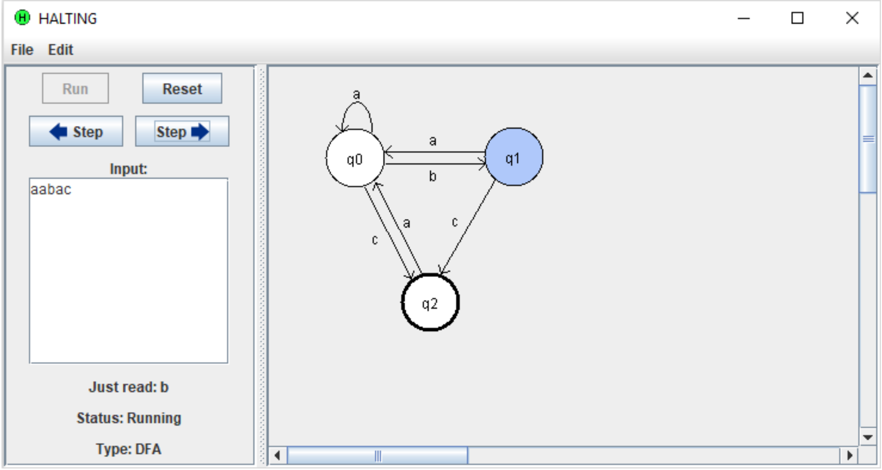
Input:
aabac

Just read: c

Status: Accepted

Type: DFA

Option 2: Select the Step Forward button (Step ⟶) to run a single input character through the automaton; the current input character will be indicated beneath the text box (“Just read: ”), and the current state will be indicated by color.



Input:

aabac

Just read: b

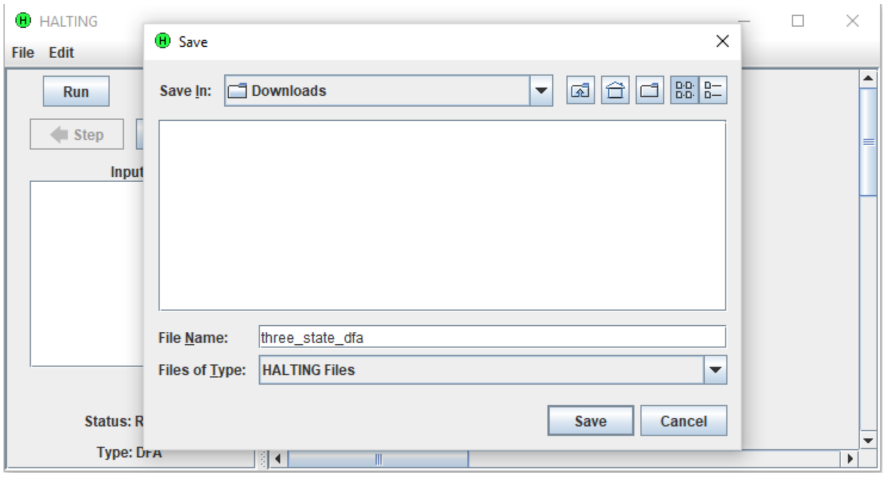
Status: Running

Type: DFA

Continue selecting Step Forward to consume the input one character at a time. Select Step Backward (Step ⟵) to take a step backwards, in effect “un-consuming” the most recently consumed input and reverting to the previous state. This process will eventually result in either an error or an indicated final state, as described in Option 1.

<u>Saving and loading an automaton:</u>

Save the current automaton by selecting File > Save. This will open a file explorer where you may select a name and location for the automaton file. HALTING automaton files have the extension .hlt. When saving an automaton, the locations of the states are preserved, meaning you don't need to rearrange the states every time you open a file.

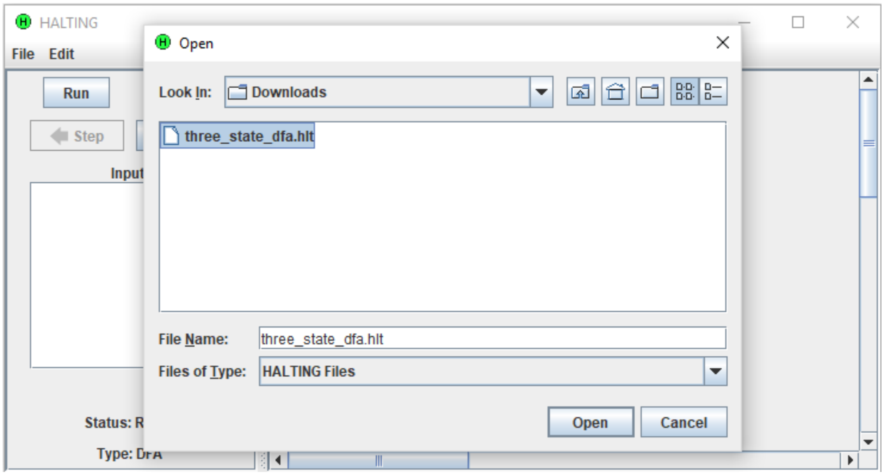


Save In: Downloads

File Name: three_state_dfa

Files of Type: HALTING Files

Load a previously saved automaton by selecting File > Open. In the file explorer that appears, locate and select the desired .hlt file. This will open that automaton in a new window, or in the current window if no automaton has been built in it yet. The states will be automatically placed in their locations from the time of the file saving.



Look In: Downloads

File Name: three_state_dfa.hlt

Files of Type: HALTING Files

Future Features

Although initially developed as an honors thesis project, HALTING will continue to improve beyond graduation. The following are some plans for future work:

- Implementation for non-deterministic machines
- User customization, e.g. font sizes, default file location
- Image exporting of automata / video exporting of automata running through input
- Movable transition arrows; curvable transition arrows
- Intelligent state auto-placement using graph theory
- Various finite machine related utilities, e.g. conversion from NFA to DFA, DFA state minimization

Problems, Questions, and Suggestions

Is something not working correctly? Do you have a feature request? Contact the developer by submitting an issue for the project’s Github at https://github.com/m-montgomery/HALTING.