Abstract Automata

Three Basic Models: Finite-State, Pushdown, and Turing

Always in One of a Finite Number of Internal States (named using capital letters, "A" is normally the initial state)

Scan One Cell of Input Tape per Machine Cycle (input symbols may be upper- or lower-case letters, digits, etc.)

Finite-State Machines
- State Transition Based on Current Input Symbol and Current State
- Stop at End of Input, Accept if in a Final State

Pushdown Machines
- Adds a Pushdown Tape to Finite-State Model
- State Transition Based on Input Symbol, State, and Top of Stack
- Top Stack Symbol can also be Replaced on Each Machine Cycle
- Stop at End of Input, Accept if in a Final State and Stack is Empty

Turing Machines
- Adds Ability to move Left as well as Right on Tape, and the Ability to Modify the Symbol on the Tape
- State Transition Based on Input Symbol and Current State

Execution Continues until Transition to Special "Halt" State (labeled "#" in the simulator)

Running Existing Machines

Viewing the State Tables
- Next/Previous Machine Buttons
- "Machine" Menu lists all Machines for Selection
- <⌘-P> to Print out Copy of Screen
- Cycling through States (PDM only)

Click Current State Box in upper lefthand corner of State Table

Sample Machine Descriptions

- Finite-State Machines
  - Odd Parity
    - Input: string of 0's & 1's
    - Accept if number of 1's is odd
  - Pattern Recognition
    - Input: word over {A,D,E,I,V}
  - Accept if input contains "DAVE" or "DAVID"
  - Addition modulo 5
    - Input: string over {0,1,2,3,4}

Accept if digits add up to 0 mod 5
- Pushdown Machines

a's = **b**'s (1 state)
Input: word over {a,b}
Accept if the number of a's = number of b's
Matched Parentheses (1 state)
Input: left and right parentheses
Accept if input is a properly parenthesized expression
$\mathbf{a^i b^i}$ (2 states)
Input: word over {a,b}
Accept if word contains a number of a's followed by same number of b's

Turing Machines
Duplicate
Input: word, w, over {a,b}
Halts with ww on tape
$\mathbf{a^i b^i c^i}$
Input: word over {a,b,c}
Halts with "Y" if word contains the same # of each letter in order
Halts with "N" otherwise
Sorting
Input: string {1,2,3}
Halts after sorting digits into ascending order
Count the **a**'s or **b**'s
Input: "|W|a" or "|W|b", where W ∈ {a,b}*
Counts the number of a's or b's (as specified)
Busy Beaver Machines
Start with read/write head in middle of blank tape
Writes as many 1's as it can before halting
Remainder Function
Input: two unary numbers (string of 1's)
following a space, separated by a space
Computes the remainder on dividing first by second

The <RUN> Button
Continue execution until end of input (FSM & PDM only)
Continue execution until Halt state is reached (TM only)
The <PAUSE> Button suspends execution (in case of infinite loops)

The <STEP> Button

Execute one machine cycle

The Input String
Click on string to change, double-click to highlight

Use normal MAC editing functions

Re-Start

Click on Read Head to Reset

Click to Clear the Stack (PDM only)

CREATING A NEW MACHINE

The <NEW> Button

Presents you with a 2-state, 2-input blank machine (FSM & TM only)

Presents you with a 2-input, 2-stack-symbol blank machine (PDM)

Enter Name for your Machine

States

State Names are restricted to capital letters
Click the <down arrow> to Add More States
Click the <up arrow> to Delete States
Shift-Click State along left edge of table to make final (FSM only)
Shift-Click State above the state table to make final (PDM only)
Click State above table to see next state table (PDM only)

Go to state # to Halt (TM only)

Input Alphabet

Click symbols on top of state table to change
Click <right arrow> attached to table to add new symbols
Click <left arrow> attached to table to delete symbols

The State Table

Click within State Table to Specify Function of Machine:
Next State (FSM)
Next State, Stacktop Replacement (PDM)
Next State, Input Tape Replacement, Direction to Move (TM)