Drawing Finite State Machines in LaTeX and TikZ
A Tutorial

Satyaki Sikdar and David Chiang
ssikdar@nd.edu

Version 3
January 17, 2018

1 Introduction

"LaTeX (pronounced lay-tek) is an open-source, multiplatform document preparation system for producing professional-looking documents....It is particularly suited to producing long, structured documents, and is very good at typesetting equations" [University of Edinburgh Information Services, 2014].

The capabilities of the system are greatly enhanced with the help of native and third-party packages.[1] TikZ[2] is a package for drawing all kinds of graphics.

This tutorial introduces the reader to LaTeX and the TikZ package, particularly for drawing state diagrams of finite automata.

2 Setting up LaTeX

To proceed with the tutorial, a working LaTeX setup is necessary. You may install it locally on your machine, but the simplest thing to do is use ShareLaTeX (sharelatex.com). If you sign up using your nd.edu address, you'll get unlimited private project. For further information regarding setup, visit http://www.latex-project.org/get/.

3 The preamble

Every LaTeX document starts with a *preamble*. To make our automata look like the ones in the textbook [Sipser, 2012], use the following preamble:

```
\documentclass{article}       % What kind of document this is
\usepackage{tikz}             % Import the tikz package
\usetikzlibrary{automata}     % Import library for drawing automata
\usetikzlibrary{positioning}  %                 ...positioning nodes
\usetikzlibrary{arrows}       %                 ...customizing arrows
\tikzset{node distance=2.5cm, % Minimum distance between two nodes. Change if necessary.
         every state/.style={ % Sets the properties for each state
           semithick,
           fill=gray!10},
         initial text={},     % No label on start arrow
         double distance=2pt, % Adjust appearance of accept states
         every edge/.style={  % Sets the properties for each transition
```

[1] The Comprehensive TeX Archive Network (CTAN) is the central place for all kinds of material around TeX. https://www.ctan.org/?lang=en

[2] https://www.ctan.org/pkg/pgf?lang=en

```
            draw,
            ->,>=stealth’,      % Makes edges directed with bold arrowheads
            auto,
            semithick}}
\let\epsilon\varepsilon
```

After the preamble comes the content:

```
\begin{document}
% Content goes here
\end{document}
```

4 Basics

While there are many tutorials online, I suggest two: University of Edinburgh Information Services [2014] and `https://www.latex-tutorial.com/`.

Here are some symbols often used in this course:

symbol	control sequence	usual meaning
Σ	`\Sigma`	alphabet
Γ	`\Gamma`	another alphabet
ε	`\varepsilon`	empty string
$\circ$	`\circ`	concatenation
`#`	`\texttt{\#}`	marker symbol
`$`	`\texttt{\$}`	marker symbol
␣	`\textvisiblespace`	blank symbol
$\{\ \}$	`\{ \}`	delimiters for sets
$\emptyset$	`\emptyset`	empty set
$\neq$	`\neq`	is not equal to
$\in$	`\in`	is an element of
$\notin$	`\notin`	is not an element of
$\subseteq$	`\subseteq`	is a subset of
$\rightarrow$	`\rightarrow`	(various meanings)
δ	`\delta`	transition function
α	`\alpha`	regular expression
*	`^\ast`	Kleene star

5 Drawing automata

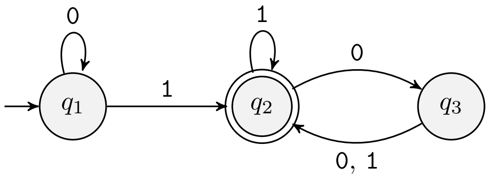
Let’s start off with a simple DFA from Sipser [2012] (Figure 1.6). The formal description of the DFA is:

$$M_1 = (\{q_1, q_2, q_3\}, \{\texttt{0}, \texttt{1}\}, \delta, q_1, \{q_2\}),$$

where δ is given by:

	`0`	`1`
q_1	q_1	q_2
q_2	q_3	q_2
q_3	q_2	q_2

Below is the code that generates the state diagram of M_1.



```
\begin{tikzpicture}
  \node[state, initial] (q1) {$q_1$};
  \node[state, accepting, right of=q1] (q2) {$q_2$};
  \node[state, right of=q2] (q3) {$q_3$};
  \draw (q1) edge[loop above] node {\tt 0} (q1);
  \draw (q1) edge node {\tt 1} (q2);
  \draw (q2) edge[loop above] node {\tt 1} (q2);
  \draw (q2) edge[bend left] node {\tt 0} (q3);
  \draw (q3) edge[bend left] node {{\tt 0}, {\tt 1}} (q2);
\end{tikzpicture}
```

Below, we'll go through this example piece by piece.

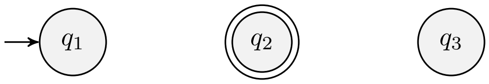
5.1 The `tikzpicture` environment

Inside the document, each TikZ diagram must reside in a `tikzpicture` environment:

```
\begin{tikzpicture}
% tikz code goes here
\end{tikzpicture}
```

5.2 Nodes

Let's start off by drawing the nodes. Nodes can be positioned either manually or relative to other nodes. Relative placement is often much easier.



```
\begin{tikzpicture}
  \node[state, initial] (q1) {$q_1$};
  \node[state, accepting, right of=q1] (q2) {$q_2$};
  \node[state, right of=q2] (q3) {$q_3$};
\end{tikzpicture}
```

The general form of the `\node` command is:

```
\node[<options>] (<name>) at (<x>,<y>) {<label>};
```

The `<options>`, `(<name>)`, and `at (<x>,<y>)` are all optional, but the `{<label>}` is required.

Options The options (for finite automata) are:

- `state`: always give this option to draw a node as a state
- `initial`: specifies the start state
- `accepting`: specifies an accept state

Note that the size of a node depends on the length of its label; to force a minimum size (say, 1 inch), use `minimum size=1in`.

Name The name of a node is the name by which you refer to the node, when positioning other nodes relative to it or when drawing edges into or out of it.

Position You specify the absolute position of a node using `at (<x>,<y>)` where `<x>` and `<y>` are coordinates (`<x>` coordinates go to the right; `<y>` coordinates go up).

Or you can specify a relative position using `left of=<name>`, `right of=<name>`, `above of=<name>`, `below of=<name>`. There's also `above left of=<name>`, etc.[3]

The `positioning` library which we have already imported provides some further options.

- `xshift=`x`, yshift=`y: Gives manual control of the node positions after relative placement. Eg:

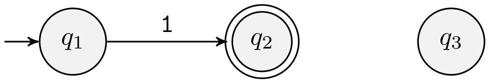
```
\node[state, right of=q1, xshift=1cm] (q2) {$q_2$};
```

Label This can be anything you want. Typically you will surround it with dollar signs to use math mode.

5.3 Edges

Once the states are all in place, let's start adding the transitions, that is, the edges between the states.

The `\draw` command can be used to draw the edges between the already created nodes (states).



```
\begin{tikzpicture}
  \node[state, initial] (q1) {$q_1$};
  \node[state, accepting, right of=q1] (q2) {$q_2$};
  \node[state, right of=q2] (q3) {$q_3$};
  \draw (q1) edge node {\tt 1} (q2);
\end{tikzpicture}
```

The general syntax is as follows:

```
\draw (<source node>) edge[<options>] node {<label>} (<dest node>);
```

Source and destination nodes Note that `<source node>` and `<dest node>` are the names of the nodes, not their labels.

Options The `<options>` modify the appearance of the edge.

- For edges that start and end in the same node (self-loops), you must `loop above`, `loop above`, `loop left`, or `loop right`.
- By default, the edges are straight, so to prevent overlaps use `bend left` or `bend right`.
- To modify the placement of the edge label, use `above` or `below`.

Label This can be anything you want. Note that Sipser uses typewriter font for symbols, so you probably want to write `{\tt 0}` or `\texttt{0}`.

Shorthand Multiple edges can be drawn with the same `draw` command, like so:

```
  \draw (q1) edge[loop above] node {\tt 0} (q1)
             edge node {\tt 1} (q2)
  \draw (q2) edge[loop above] node {\tt 1} (q2)
             edge[bend left] node {\tt 0} (q3)
  \draw (q3) edge[bend left] node {{\tt 0}, {\tt 1}} (q2);
```

[3] Technically, these options are deprecated, but we find them useful anyway. See `https://tex.stackexchange.com/questions/9386/difference-between-right-of-and-right-of-in-pgf-tikz`.

6 More examples

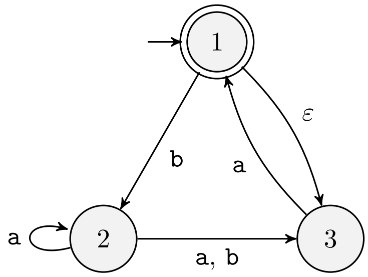
As another example, let's draw a NFA (Sipser, Figure 1.42).

$$N_1 = (\{1,2,3\}, \{\mathtt{a}, \mathtt{b}\}, \delta, 1, \{1\}),$$

where δ is given by:

	a	b	ε
1	$\{\}$	$\{2\}$	$\{3\}$
2	$\{2,3\}$	$\{3\}$	$\{\}$
3	$\{1\}$	$\{\}$	$\{\}$

Below is the code that generates the state diagram of N_1.



```
\begin{tikzpicture}
  \node[state, initial, accepting] (1) at (1.5,2.6) {$1$};
  \node[state] (2) at (0,0) {$2$};
  \node[state] (3) at (3,0) {$3$};

  \draw (1) edge node{\tt b} (2)
            edge[bend left=15] node {$\epsilon$} (3)
        (2) edge[loop left] node{\tt a} (2)
            edge[below] node{{\tt a}, {\tt b}} (3)
        (3) edge[bend left=15] node{\tt a} (1);
\end{tikzpicture}
```

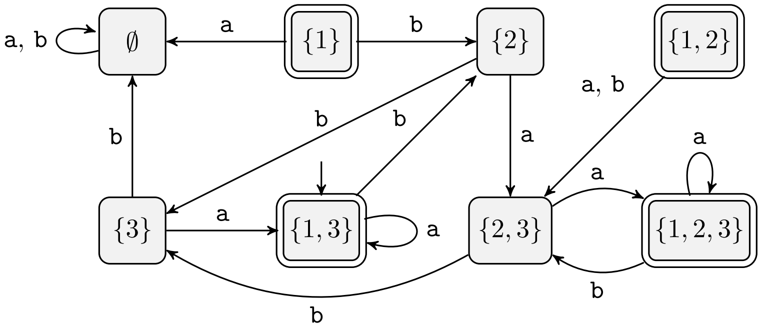
Our final example is the state diagram of the DFA equivalent to the NFA N_1:

$$D_2 = (\{\emptyset, \{1\}, \{2\}, \{3\}, \{1,2\}, \{1,3\}, \{2,3\}, \{1,2,3\}\},\ \{\mathtt{a}, \mathtt{b}\},\ \delta, \{1,3\}, \{\{1\}, \{1,2\}, \{1,3\}, \{1,2,3\}\}),$$

where δ is given by

	a	b
$\emptyset$	$\emptyset$	$\emptyset$
$\{1\}$	$\emptyset$	$\{2\}$
$\{2\}$	$\{2,3\}$	$\{3\}$
$\{3\}$	$\{1,3\}$	$\emptyset$
$\{1,2\}$	$\{2,3\}$	$\{2,3\}$
$\{1,3\}$	$\{1,3\}$	$\{2\}$
$\{2,3\}$	$\{1,2,3\}$	$\{3\}$
$\{1,2,3\}$	$\{1,2,3\}$	$\{2,3\}$

Below is the code that generates the state diagram.



```
\begin{tikzpicture}
    \tikzset{every state/.append style={rectangle, rounded corners}}
    \node[state] (emp) {$\emptyset$};
    \node[state, accepting, right of=emp] (1) {$\{1\}$};
    \node[state, right of=1] (2) {$\{2\}$};
    \node[state, accepting, right of=2] (12) {$\{1, 2\}$};
    \node[state, below of=emp] (3) {$\{3\}$};
    \node[state, initial, initial where=above, accepting, right of=3] (13) {$\{1, 3\}$};
    \node[state, right of=13] (23) {$\{2, 3\}$};
    \node[state, accepting, right of=23] (123) {$\{1, 2, 3\}$};

    \draw (emp) edge[loop left] node {{\tt a}, {\tt b}} (emp)
          (1)   edge[above] node {\tt a} (emp)
          (1)   edge node {\tt b} (2)
          (2)   edge node {\tt a} (23)
          (2)   edge[above] node {\tt b} (3)
          (12)  edge[auto=right,near start] node {{\tt a}, {\tt b}} (23)
          (3)   edge node {\tt b} (emp)
          (3)   edge node {\tt a} (13)
          (13)  edge[loop right] node {\tt a} (13)
          (13)  edge node {\tt b} (2)
          (23)  edge[bend left,above] node {\tt a} (123)
          (23)  edge[bend left] node {\tt b} (3)
          (123) edge[loop above] node {\tt a} (123)
          (123) edge[bend left,below] node {\tt b} (23);
\end{tikzpicture}
```

References

Michael Sipser. *Introduction to the Theory of Computation.* Cengage Learning, 3rd edition, 2012.

University of Edinburgh Information Services. LaTeX for beginners. http://www.docs.is.ed.ac.uk/skills/documents/3722/3722-2014.pdf, 2014.