USING XY TO TYPESET AUTOMATA

ARUN DEBRAY
JULY 5, 2016

Contents

1. Introduction

Hello! If you're reading this, you're probably a student in an automata theory class, such as Stanford's CS 103 or CS 154, and you want to learn how to typeset automata in LaTeX. Many people use a library called `tikz` to do this, and it works well, but it has its own issues: it is powerful but complicated to learn, and making a diagram takes longer. To me, the XY package achieves the right balance between power and ease of use.

. . . That said, I have yet to see a guide for XY that makes it easy to learn. [2] is pretty comprehensive, but not very clear, especially since it has only one example of an automaton and doesn't explain it. So in this guide, I will

(1) teach you enough of XY to typeset good-looking automata for course notes or homework, and
(2) provide explanations, rather than just examples, to minimize confusion and allow you to spend less time on the mechanics of typesetting automata.

I will assume that you're familiar with the basics of LaTeX. There are many good places to learn this, such as [1].

To use XY, place

```
\usepackage[all]{xy}
```

in the preamble of your document. The rest of this document assumes that you've done this.

For the forseeable future, the latest version of this guide will be posted at https://ma.utexas.edu/users/a.debray/lecture_notes/using_xy.pdf.

2. Simple Diagrams

XY is usually used to create commutative diagrams, which mathematicians use to display functions between objects. I'll use a few examples to demonstrate how XY works.

Here are the basic principles of XY.

- Items are typeset in a grid, much like a table, array, or matrix environment: `&` separates columns and `\\` separates rows.
- The grid is placed inside the command `\xymatrix{...}`, which must be called in math mode.

- The basic diagram command is an arrow, `\ar[`*destination*`]`. The command is issued at the arrow's origin, and the destination is specified with directions: `\ar[d]` points the arrow one row downward, `\ar[r]` one column to the right, `\ar[u]` points it one row upwards, and `\ar[l]` points it one column to the left.

For example, here's a simple diagram:

```
\[\xymatrix{
    A \ar[r] \ar[d] & B \ar[d] \\
    C               & D. \ar[l]
}\]
```

Output:

$$\begin{array}{ccc} A & \longrightarrow & B \\ \downarrow & & \downarrow \\ C & \longleftarrow & D. \end{array}$$

Combining directions. `\ar[dl]` typesets a diagonal arrow going down and to the left, and `\ar[rr]` travels two grid entries to the right. For example:

```
\[\xymatrix{
    \Psi_0 \ar[r] \ar[d] & B \ar[rr] \ar[dr] & & C \\
    \ar[r] & 3 \ar[r] & q_1 \ar[ur] & 5
}\]
```

Output:

$$\begin{array}{ccccc} \Psi_0 & \longrightarrow & B & \longrightarrow & C \\ \downarrow & & & \searrow & \uparrow \\ \longrightarrow & 3 & \longrightarrow & q_1 & 5 \end{array}$$

Labeling arrows. Looking along the direction of the arrow,

- `\ar[r]^{`*label*`}` places the label on the left.
- `\ar[r]_{`*label*`}` places the label on the right.
- `\ar[r]|{`*label*`}` places the label on top of the arrow.

For example, in the following diagram, all 0s are placed with `^` and all 1s are placed with `_`.

$$\begin{array}{ccccc} A & \overset{0}{\underset{1}{\longrightarrow}} & B & \xrightarrow{\vartheta(x)} & C \\ {\scriptstyle 0}\uparrow{\scriptstyle 1} & & {\scriptstyle 1}\downarrow{\scriptstyle 0} & & \downarrow{\scriptstyle ab^*} \\ D & \underset{0}{\overset{1}{\longleftarrow}} & E & & F \end{array}$$

Source code:

```
\[\xymatrix{
    A \ar[r]^0_1 & B \ar[d]^0_1 \ar[r]_{\vartheta(x)} & C \ar[d]|{ab^*}\\
    D \ar[u]^0_1 &E \ar[l]^0_1 &F
}\]
```

Exercise 2.1. Try typesetting

$$\begin{array}{ccccc} q_0 & \overset{a_1}{\longrightarrow} & q_1 & & q_2. \\ & \searrow^{a_2} & & \nearrow_{a_3} & \\ & & q_3 & & \end{array}$$

3. Automata

So this is all excellent, you say, but you're here to learn how to typeset automata. How can we turn these diagrams into automata? There's three modifications to make, and we'll treat each in turn.

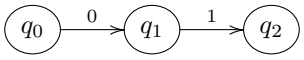
Circling states. Each state in an automaton is bordered by a circle. In XY, one can wrap a grid entry in a circle by prefacing it with `++[o][F]`, and writing

```
\entrymodifiers={++[o][F-]}
```

before the `\xymatrix{...}` command draws a circle around every entry, e.g.

```
\[\entrymodifiers={++[o][F-]}\xymatrix{
    q_0 \ar[r]^0 & q_1 \ar[r]^1 & q_2
}\]
```

Output:



I usually make this into a macro

```
\newcommand{\atm}{\entrymodifiers={++[o][F-]}}
```

which is used as

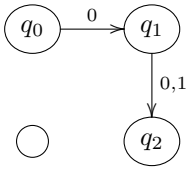
```
\atm\xymatrix{...}
```

which is less cluttered. I'm going to follow this convention for the rest of the guide.

`\entrymodifiers` modifies every entry — even empty ones. This means if you use empty states in automata, they'll have undesirable small circles:

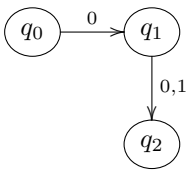
```
\[\atm\xymatrix{
    q_0 \ar[r]^0 & q_1\ar[d]^{0,1}\\
    & q_2
}\]
```

The lower left entry is empty, but it gets surrounded by a circle anyways.



To fix this, "reset" this state with `*{}`. Here's what the correct code looks like:

```
\[\atm\xymatrix{
    q_0 \ar[r]^0 & q_1 \ar[d]^{0,1}\\
    *{} & q_2
}\]
```

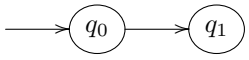


Much nicer.

Start state. The start state of an automaton is marked by an arrow pointing to it. In XY, you add an empty state to the left with an arrow pointing to the start state, and use `*{}` to make sure it doesn't get a circle drawn around it. Here's an example.

```
\[\atm\xymatrix{
    *{} \ar[r] & q_0 \ar[r] & q_1
}\]
```

Output:



Accept states. Finally, accept states are denoted with a double circle. The incantation for this is

`*++[o][F=]`

- `*` overwrites the behavior of `\entrymodifiers`, and
- `[F=]` draws a double circle (`[F-]` makes a single circle).

I also make this into a macro

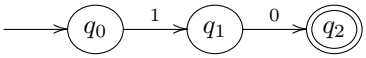
```
\newcommand{\accept}[1]{*++[o][F=]{#1}}
```

for ease of typing and readability; I will follow this convention for the rest of the guide.

Here's how to use this macro:

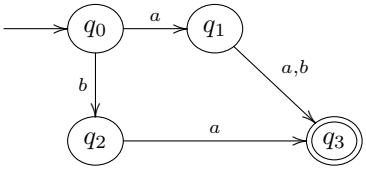
```
\[\atm\xymatrix{
    *{} \ar[r] & q_0 \ar[r]^1 & q_1 \ar[r]^0 &\accept{q_2}
}\]
```

Output:



At this point, you have the techniques to typeset many simple automata.

Exercise 3.1. Try typesetting



4. More Interesting Arrows

The real power of XY-pic, and what makes it useful for typesetting automata, is what all it can do with arrows.

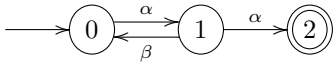
Back-and-forth arrows. The syntax

`\ar@<0.1cm>[r]`

slides this arrow 0.1 cm to the left (the `^` direction), and similarly `\ar@<-0.1cm>[r]` slides it to the right (the `_` direction). Use this to typeset arrows going back and forth between two states:

```
\[\atm\xymatrix{
    *{} \ar[r] & 0 \ar@<0.1cm>[r]^\alpha & 1 \ar@<0.1cm>[l]^\beta \ar[r]^\
       alpha & \accept{2}
}\]
```

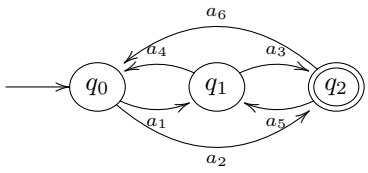
Output:



Bending arrows. The syntax

`\ar@/^0.2cm/[r]`

bends this arrow 0.2 cm in the `^` direction (left). Similarly, `\ar@/_0.2cm/[r]` bends this arrow 0.2 cm in the `_` direction (right). Example:



Source code:

```
\[\atm\xymatrix{
    *{} \ar[r] & q_0\ar@/_0.3cm/[r]_{a_1} \ar@/_0.8cm/[rr]_{a_2}
               & q_1\ar@/^0.3cm/[r]^{a_3} \ar@/_0.3cm/[l]_{a_4}
               & \accept{q_2}\ar@/^0.3cm/[l]^{a_5} \ar@/_0.8cm/[ll]_{a_6}
}\]
```

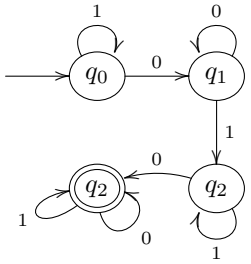
Self-transitions. When a state has a transition to itself, use the syntax

`\ar@(`*in*, *out*`)`

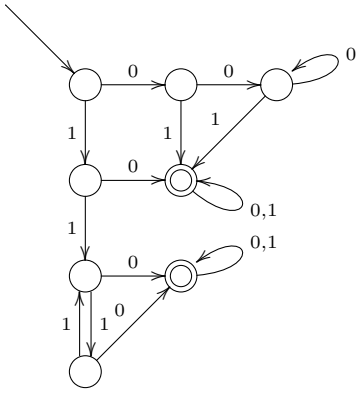
where *in* and *out* are one of the directions `u` (up), `ur` (upper right), `r` (right), `dr` (lower right), `d` (down), `dl` (lower left), `l` (left), and `ul` (upper left). This typesets an arrow from the *in* direction to the *out* direction.

```
\[\atm\xymatrix{
    *{} \ar[r] & q_0 \ar@(ul,ur)^1 \ar[r]^0
        & q_1\ar[d]^1 \ar@(ur,ul)_0\\
    *{} & \accept{q_2}\ar@(dl,l)^1 \ar@(d,r)_0
        & q_2\ar@/_0.2cm/[l]_0 \ar@(dr,dl)^1
}\]
```

Output:



Exercise 4.1.

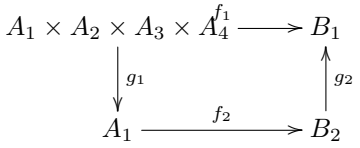


There are many, many things that XY can do with arrows, but these are the most useful ones for automata. Consult [2] for some more.

5. More Interesting Labels

This section isn't super important, but sometimes can improve the appearance of diagrams.

Placement of labels along arrows. By default, labels are placed halfway between grid points, not halfway along arrows. Sometimes this is undesirable:



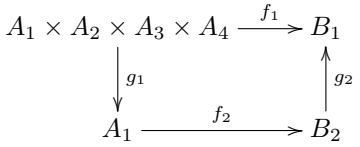
Source code:

```
\[\xymatrix{
    A_1 \times A_2\times A_3\times A_4\ar[r]^{f_1}\ar[d]^{g_1} & B_1\\
    A_1\ar[r]^{f_2} & B_2\ar[u]_{g_2}
}\]
```

There are a few ways to fix this.

- Replacing `^` or `_` with `^-` or `_-` repositions the label at the center of the arrow:

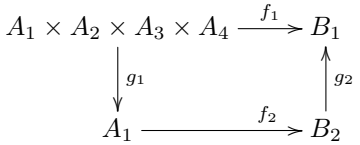
```
\[\xymatrix{
    A_1 \times A_2\times A_3\times A_4\ar[r]^-{f_1}\ar[d]^{g_1} & B
        _1\\
    A_1\ar[r]^{f_2} & B_2\ar[u]_{g_2}
}\]
```



- `^(`a`)` indicates that the label should be a of the way from the center of the source `(0)` to the center of the destination `(1)`. For example, `(0.3)` will place the label three-tenths of the way along the grid. Example:

```
\[\xymatrix{
    A_1 \times A_2\times A_3\times A_4\ar[r]^-{f_1}\ar[d]^{g_1} & B
        _1\\
    A_1\ar[r]^(0.72){f_2} & B_2\ar[u]_{g_2}
}\]
```

Output:



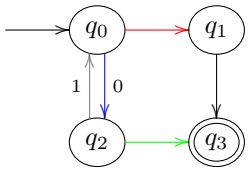
6. Color

This section assumes the `xcolor` package has been loaded.

Coloring arrows. The syntax

`\ar@[blue][r]`

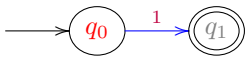
colors this arrow blue. Examples:




```
\[\atm\xymatrix{
    *{}\ar[r] & q_0 \ar@[red][r] \ar@<0.1cm>@[blue][d]^0 & q_1\ar[d]\\
    *{} & q_2\ar@<0.1cm>@[gray][u]^1 \ar@[green][r] & \accept{q_3}
}\]
```

Warning: color breaks on curved arrows, so avoid using it there.

Coloring text of states or labels. This is just like normal LaTeX coloring: use `\textcolor{`*color*`}{`*text*`}`. Example:




```
\[\atm\xymatrix{
    *{} \ar[r] & \textcolor{red}{q_0} \ar@[blue][r]^{\textcolor{purple
       }{1}} & \accept{\textcolor{gray}{q_1}}
}\]
```

Coloring frames. Recall that the default frame was defined as `++[o][F-]`; to add color, use

`++[o][F-:`*color*`]`

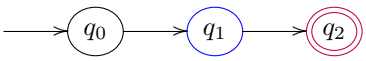
and for accept states, use

`++[o][F=:`*color*`]`

which uses the double circle.

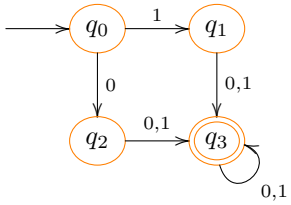
To color a single state, reset it from the default modifier by using `*++[o][F-:`*color*`]`:

```
\[\atm\xymatrix{
    *{} \ar[r] & q_0\ar[r] & *++[o][F-:blue]{q_1} \ar[r] &*++[o][F=:purple
       ]{q_2}
}\]
```



To color every state, use `\entrymodifiers`:

```
\[\entrymodifiers={++[o][F-:orange]}\xymatrix{
    *{} \ar[r] & q_0\ar[d]^0 \ar[r]^1 & q_1\ar[d]^{0,1}\\
    *{} & q_2\ar[r]^{0,1} & *++[o][F=:orange]{q_3} \ar@(d,r)_{0,1}
}\]
```

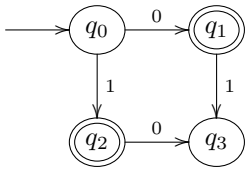


If you're going to do this for a lot of automata, it might be worth defining macros for these.

7. Other Tips

Equation numbering. By default, equation numbers are placed at the top of the equation:

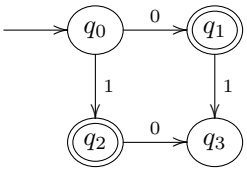
(1)



If you prefer for it to be centered, wrap the `\xymatrix` command in a `gathered` environment:

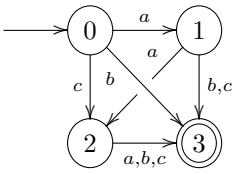
```
\begin{equation}
\begin{gathered}
\atm\xymatrix{
    *{} \ar[r] & q_0 \ar[r]^0 \ar[d]^1 & \accept{q_1} \ar[d]^1\\
    *{} & \accept{q_2} \ar[r]^0 & q_3
}
\end{gathered}
\end{equation}
```

(2)



Overlapping arrows. `\ar[r]|\hole` puts a hole in the arrow, whose location you can modify with `-` or `()` as usual. This is useful for making overlapping arrows easier to read.

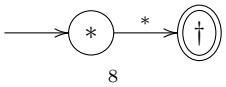
```
\[\atm\xymatrix{
    *{} \ar[r] & 0 \ar[r]^a \ar[dr]_(0,3)b \ar[d]_c & 1 \ar[dl]|\hole
       _(0.3)a \ar[d]^{b,c}\\
    *{} & 2 \ar[r]_{a,b,c} & \accept{3}
}\]
```



Special characters. `*` has meaning to XY, so to label a state or arrow with it, use `{*}`; likewise for other special characters.

```
\[\atm\xymatrix{
    *{} \ar[r] & {*} \ar[r]^{*} & \accept{\dagger}
}\]
```

Output:



8. Conclusion

There is much more to XY-pic than contained in this guide, but this should be sufficient to typeset automata. I hope it was useful.

If you have questions, comments, or concerns, or you find mistakes or typos, please contact me at a.debray@math.utexas.edu.

One thing I'd like to add is an appendix on common errors in XY; LaTeX doesn't have the clearest error messages, and XY unfortunately upholds this tradition.

Appendix A: Solutions to the Exercises

Exercise 2.1:

```
\[\xymatrix{
    q_0 \ar[r]^{a_1} \ar[dr]^{a_2} & q_1 & q_2. \\
    & q_3 \ar[ur]_{a_3}
}\]
```

Exercise 3.1:

```
\[\atm\xymatrix{
    *{}\ar[r] & q_0\ar[r]^a\ar[d]_b &q_1\ar[dr]^{a,b}\\
    *{} & q_2\ar[rr]^a &*{} & \accept{q_3}
}\]
```

Exercise 4.1:

```
\[\atm\xymatrix{
    *{}\ar[dr]\\
    *{}& \ar[r]^0\ar[d]_1 &\ar[r]^0\ar[d]_1 & \ar@(r,ur)_0\ar[dl]_1\\
    *{} & \ar[r]^0\ar[d]_1 & \accept{}\ar@(dr,r)_{0,1}\\
    *{} & \ar[r]^0\ar@<0.5ex>[d]^1& \accept{}\ar@(r,ur)_{0,1}\\
    *{} & \ar@<0.5ex>[u]^1 \ar[ur]^0
}\]
```

References

[1] "Wikibooks: LaTeX." https://en.wikibooks.org/wiki/LaTeX. 24 October 2013.

[2] Rose, Kristoffer H. "XY-pic User's Guide." https://www.math.ubc.ca/~cautis/tools/xypic.pdf. 24 May 2012.

[3] Rose, Kristoffer H., and Ross Moore. "XY-pic Reference Manual." https://cs.brown.edu/about/system/software/latex/doc/xyrefer.pdf. 16 February 1999.