Tegnekurs i TikZ

Veronika Heimsbakk
veronahe@ulrik.uio.no

2. oktober 2014

Innhold

Figurer

1 The Basics

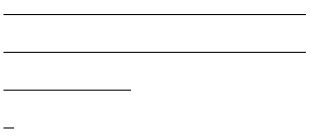
For å kunne bruke pakken TikZ må man først inkludere `\usepackage{tikz}` i dokumentet sitt.

1.1 `tikzpicture`

Alle illustrasjoner som skal tegnes ved hjelp av pakken TikZ krever et miljø som heter `tikzpicture`.

```
\begin{tikzpicture}
    <kode her>
\end{tikzpicture}
```

1.2 Linjer

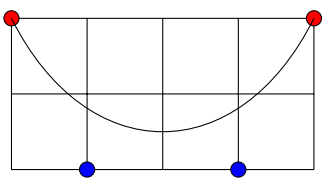


En av de mest brukte TikZ kommandoene er `\draw`. For å tegne ei rett linje sier man hvor man vil tegne *fra* og *til*:

```
\draw (0,2) -- (4,2);
\draw (0cm,1.5cm) -- (4cm,1.5cm);
\draw (0em, 1cm) -- (4em, 1cm);
\draw (0pt, 0.5cm) -- (4pt, 0.5cm);
```

1.3 Kurver

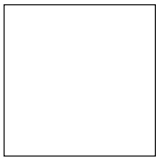
Vi bruker kontrollpunkter for å lage en kurvet linje. I eksempelet her, så starter vi i koordinatene `(-2,2)` og så tegner vi en kurve til første kontrollpunkt som er `(-1,0)`, så videre til `(1,0)`, og til slutt ender kurven opp i slutt-punktet som er `(2,2)`.



```
\draw (-2,2) .. controls (-1,0) and (1,0) .. (2,2);
```

1.4 Kvadrat

Videre kan vi bygge på og lage et kvadrat:

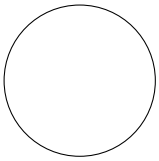


```
\draw (0,0) -- (2,0) -- (2,2) -- (0,2) -- (0,0);
```

Vi kan også bruke nøkkelordet `rectangle`, og lage en kortversjon som gjør akkurat det samme:

```
\draw (0,0) rectangle (2,2);
```

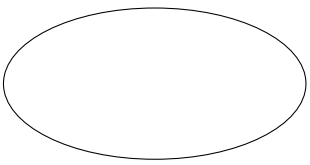
1.5 Sirkel



Den første koordinaten er sirkelens sentrum, og lengden vi oppgir til slutt er sirkelens radius.

```
\draw (2,2) circle (1cm);
```

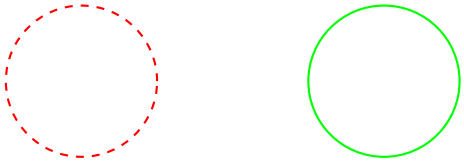
Ellipser tegnes ved at vi oppgir radiusen i x- og y-retningene:



```
\draw (2,2) ellipse (2cm and 1cm);
```

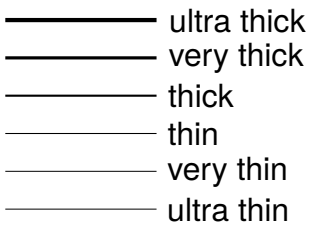
1.6 Pynte litt

For å pynte litt på sirkelen vår, kan vi legge til noen ekstra argumenter til `\draw`-kommandoen. For eksempel slik:



```
\draw[red, thick, dashed] (2,2) circle (1cm);
\draw[green, thick] (6,2) circle (1cm);
```

1.7 Tykkelser



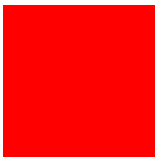
Figur 1: Mulige tykkelser i TikZ.

1.8 Farger

Figur 2: Mulige farger i TikZ.

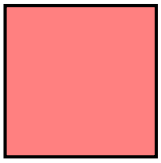
1.9 Bruke farger

Vi kan også fylle formene våre ved å bruke kommandoen `\fill`.



```
\fill[red] (0,0) rectangle (2,2);
```

Om vi ønsker å legge til en kant rundt kvadratet, kan vi bruke kommandoen `\filldraw`. Her fyller vi kvadratet rødt og hvitt, 50/50.



```
\filldraw[red!50, draw=black, very thick] (0,0) rectangle (2,2);
```

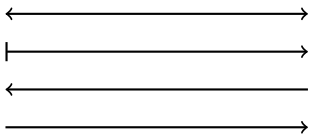
1.9.1 Gradient

Vi har også gradient i TikZ, og det kan se slik ut:

```
\shade[left color=black, right color=red]  (0,0) rectangle (2,2);
\shade[top color=black, bottom color=red]  (0,0) rectangle (2,2);
\shade[inner color=black, outer color=red] (0,0) rectangle (2,2);
```

1.10 Plotte funksjoner

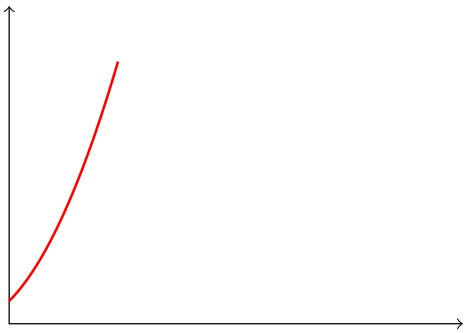
Man kan også plotte funksjoner i TikZ. Da er det kjekt å kjenne til de forskjellige typer piler.



Figur 3: Forskjellige piler i TikZ.

```
\draw[<->] (0,1.5) -- (4,1.5);
\draw[|->] (0,1) -- (4,1);
\draw[<-]  (0,0.5) -- (4,0.5);
\draw[->]  (0,0) -- (4,0);
```

1.10.1 `plot`

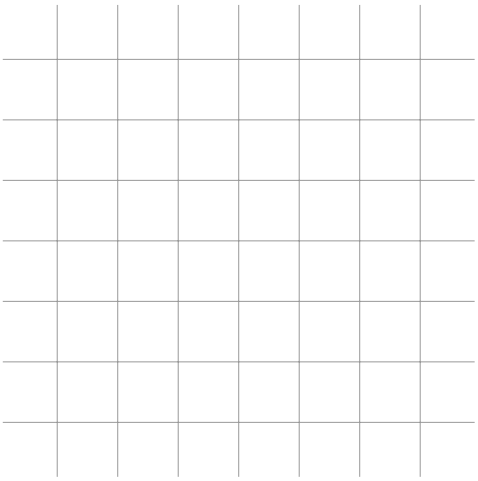


```
\begin{tikzpicture}
    \draw[<->] (0,3.5) -- (0,0) -- (5,0);
    \draw[red, thick, domain=0:1.2] plot (\x, {0.25+\x+\x*\x});
\end{tikzpicture}
```

`domain` er rekkevidden av x som blir plottet. I dette tilfellet plotter vi funksjonen $0.25 + x + x^2$. Legg merke til at det er parenteser rundt funksjonen som vi skal plotte `plot (\x, {function})`.

2 Koordinatsystem

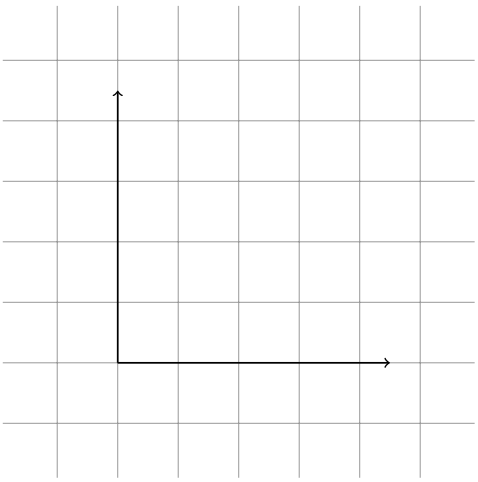
Dette eksempelet krever et rutenett, piler, noder og plassering av tall og bokstaver. Vi starter med et rutenett:



```
\draw[step=1cm,gray!80,very thin] (-1.9,-1.9) grid (5.9,5.9);
```

2.1 Piler/akser

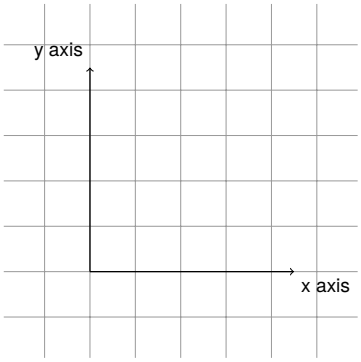
Videre trenger vi x-aksen og y-aksen. Dette er to linjer med piler i enden.



```
\draw[thick, ->] (0,0) -- (4.5,0);
\draw[thick, ->] (0,0) -- (0,4.5);
```

2.2 Noder

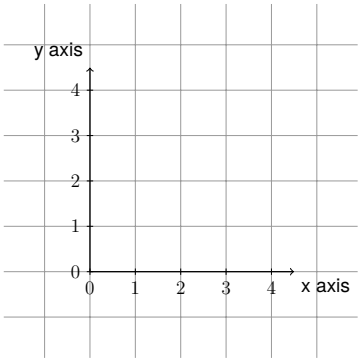
Vi kan legge på tekst (*label*) ved å bruke nøkkelordet `node`. Vi plasserer teksten ved linjene vi har tegnet ved å fortelle noden hvor vi vil ha den.




```
\draw[thick, ->] (0,0) -- (4.5,0) node[below right] {x axis};
\draw[thick, ->] (0,0) -- (0,4.5) node[above left] {y axis};
```

2.3 Løkker

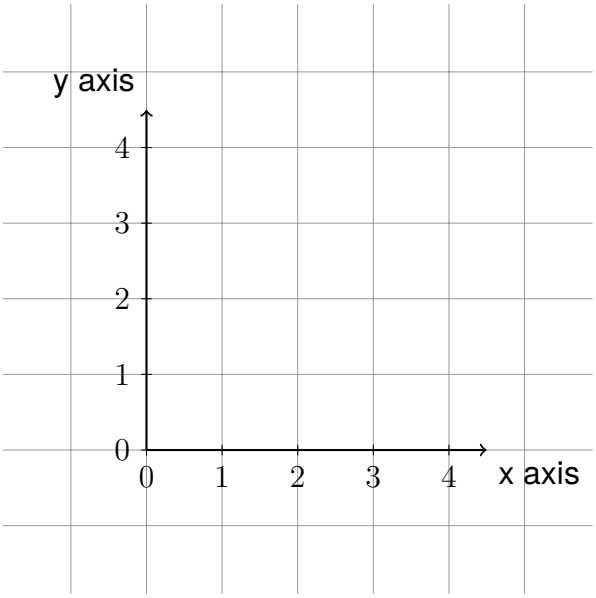
Vi kan fortsette med tallene som skal gå langs aksene ved å bruke løkker:



Denne løkken går over linjene vi allerede har tegnet, og setter en liten strek for hver centimeter. Og ved siden av linjen skriver vi et tall.

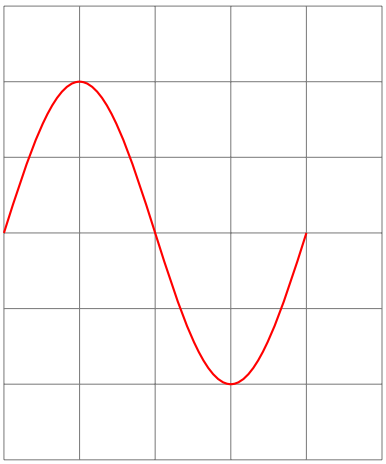
```
\foreach \x in {0,1,2,3,4}
  \draw (\x cm, 2pt) -- (\x cm, -2pt) node[below] {$\x$};
\foreach \y in {0,1,2,3,4}
  \draw (2pt, \y cm) -- (-2pt, \y cm) node[left]  {$\y$};
```

2.4 Hele koden for koordinatsystemet

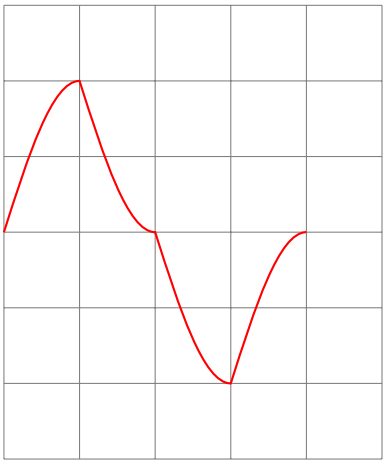



```
\begin{tikzpicture}
  \draw[step=1cm,gray!80,very thin] (-1.9,-1.9) grid (5.9,5.9);
  \draw[thick, ->] (0,0) -- (4.5,0) node[below right] {x axis};
  \draw[thick, ->] (0,0) -- (0,4.5) node[above left] {y axis};

  \foreach \x in {0,1,2,3,4}
    \draw (\x cm, 2pt) -- (\x cm, -2pt) node[below] {$\x$};
  \foreach \y in {0,1,2,3,4}
    \draw (2pt, \y cm) -- (-2pt, \y cm) node[left] {$\y$};
\end{tikzpicture}
```

2.5 Eksempler med `sin` og `cos`



```
\begin{tikzpicture}
   \draw[gray] (0,-3) grid (5,3);
   \draw[red]  (0,0) sin (1,2) cos (2,0) sin (3,-2) cos (4,0);
\end{tikzpicture}
```



```
\begin{tikzpicture}
   \draw[gray] (0,-3) grid (5,3);
   \draw[red] (0,0) sin (1,2) sin (2,0) sin (3,-2) sin (4,0);
\end{tikzpicture}
```

3 Trær

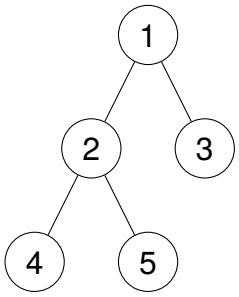
Et tre består av en rekke noder. Når vi tegner trær i TikZ starter vi med å definere rot-noden. Legg merke til attributtene vi git `tikzpicture`. Her sier vi at `every node` skal ha *stilen* (`.style`) sirkel med sort strek.


```
\begin{tikzpicture}[every node/.style={circle, draw=black}]
    \node {1};
\end{tikzpicture}
```

3.1 Bygge treet

Treet bygger vi ved å legge til barna. Barna skrives på formen:

```
child { node[opt.] {value} }
```

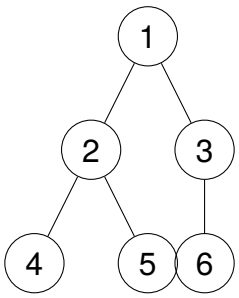



```
\node {1}
    child { node {2}
        child { node {4} }
        child { node {5} }
    }
    child { node {3} }
;
```

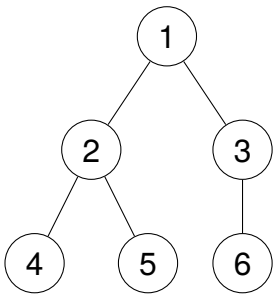
3.2 Justere avstand mellom noder

Når vi nå vil bygge videre og legge til tallet 6 under `child {node {3}}` vil vi krasje bort i 5. Da trenger vi å justere avstanden mellom søsken-noder.

Vi har:



Vi vil ha:



Da legger vi på et attributt til i listen til `tikzpicture` som forteller noe om avstanden mellom nodene.

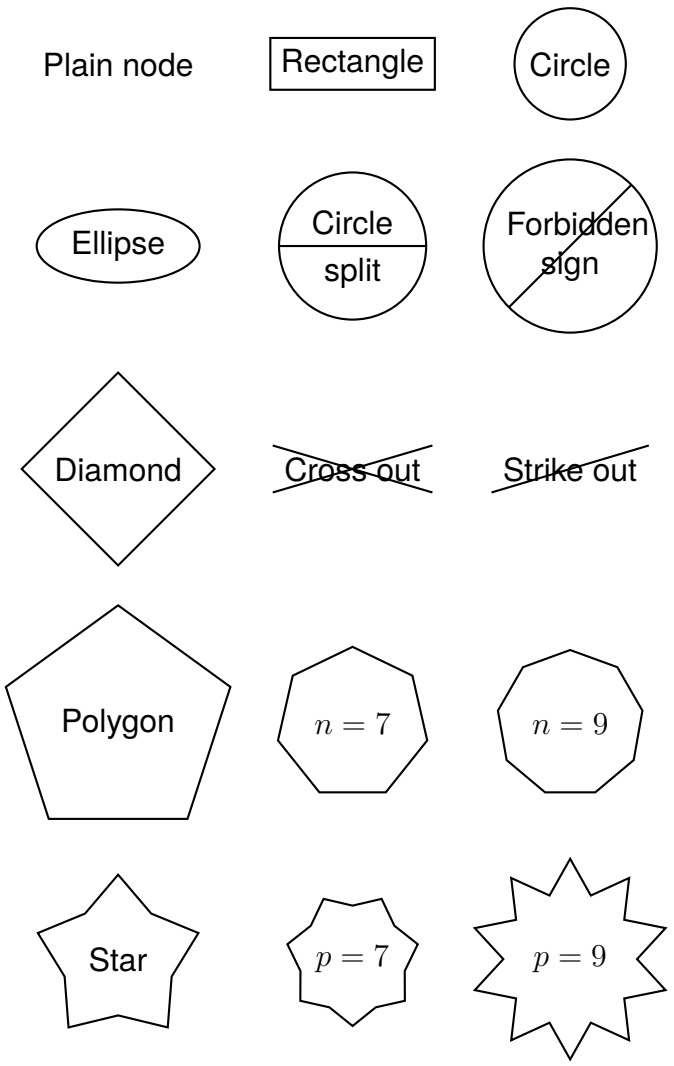
```
\begin{tikzpicture}[every node/.style={circle, draw=black},
                    level 1/.style={sibling distance=20mm},
                    level 2/.style={sibling distance=15mm}]

\node {1}
    child { node {2}
        child { node {4} }
        child { node {5} }
    }
    child { node {3}
        child {node {6} }
    }
;
\end{tikzpicture}
```

Her forteller vi at stilen til nodene på `level 1` skal være at de har avstand til sine søsken med 20 mm, og 15 mm for `level 2`. Vi kunne også lagt til attributtet `level distance` for å få større eller mindre avstand mellom lagene.

3.3 Former som noder kan ha

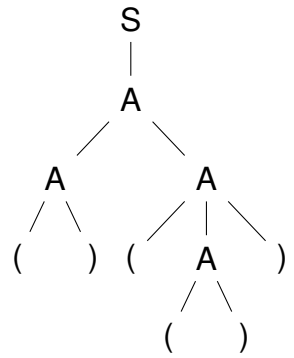
Man kan få forskjellige fasonger på noder ved å inkludere `\usetikzlibrary{shapes}`. Her er en oversikt over forskjellige fasonger en node kan ha. For å få ønsket fasong skriver man noden på denne formen:

```
\node[rectangle] {Rectangle};
\node[regular polygon, regular polygon sides=5] {n=5};
\node[circle split] {Circle \nodepart{lower} split};
```



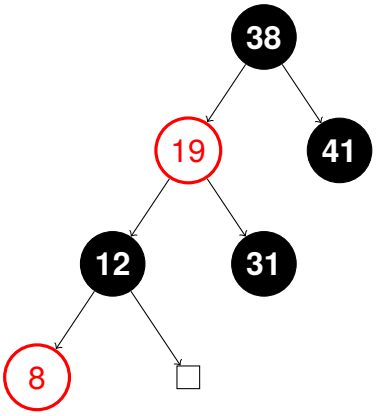
Figur 4: Forskjellige fasonger på noder.

3.4 Eksempel på et tre med avstander




```
\begin{tikzpicture}[every node/.style={},
                    level 2/.style={sibling distance=20mm},
                    level 3/.style={sibling distance=10mm},
                    level distance=30pt]
\node {S}
    child { node{A}
        child { node {A}
            child { node {(} }
            child { node {)} }
        }
        child { node {A}
            child { node {(} }
            child { node {A}
                child { node {(} }
                child { node {)} }
            }
            child { node {)} }
        }
    }
;
\end{tikzpicture}
```

3.5 Rød-svarte trær



Å tegne trær på denne måten krever ingen tilleggsbiblioteker fra TikZ. Dette er et eksempel på tegning med egendefinerte noder. Dette gjør vi via `tikzset`, her kan vi gi stilen de forskjellige typer noder.

```
\tikzset{
   treenode/.style = {align=center, inner sep=0pt},

   % Sorte noder
   node_black/.style = {treenode, circle, white,
                        font=\bfseries, draw=black,
                        fill=black, text width=0.8cm},
   % Røde noder
   node_red/.style = {treenode, circle, red, draw=red,
                      text width=0.8cm, very thick},
   % Null-pekere
   node_null/.style = {treenode, rectangle, draw=black,
                       minimum width=0.3cm, minimum height=0.3cm}
}
```

Starter med å definere `treenode`, som er felles for alle nodene. Røde og sorte noder tegnes som `circle`, hvor sorte noder har `fill=black` og tekstfarge `white`, mens røde noder har rødt omriss med `draw=red`, og tekstfarge `red`. Null-nodene sier vi skal være sorte `rectangle`. Tegnes som små kvadrater på 0.3 cm $\times$ 0.3 cm.

3.6 Bygge det rød-svarte treet

```
\begin{tikzpicture}[->,level/.style={ sibling distance = 2cm,
                    level distance = 1.5cm }]
\node [node_black] {38}
    child { node [node_red] {19}
        child { node [node_black] {12}
             child { node [node_red] {8} }
             child { node [node_null] {} }
        }
        child { node [node_black] {31} }
    }
    child { node [node_black] {41} }
;
\end{tikzpicture}
```

Setter forskjellige opsjoner med:

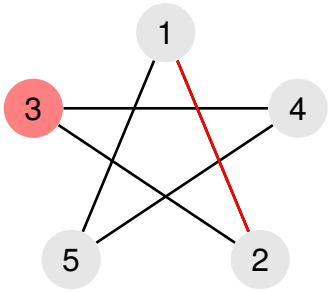
```
\begin{tikzpicture}[->,level/.style={ sibling distance = 2cm,
                    level distance = 1.5cm }]
```

Her sier vi at treet skal tegnes med piler (`->`), og at stilen (`.style`) for distansen mellom søskennoder skal være 2 cm, og distansen mellom barn og foreldre skal være 1.5 cm.

Videre så forteller vi barna i treet hva slags node de skal være.

```
child { node [node_red]   {x} }
child { node [node_black] {y} }
child { node [node_null]  {z} }
```

4 Grafer



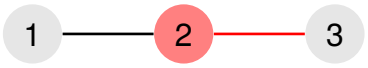
Det fins enklere måter å tegne grafer på enn dette, men jeg syns denne måten er fin. Den krever heller ingen andre biblioteker eller pakker enn TikZ selv.

Vi starter med å definere de forskjellige elementene til en graf.

```
\begin{tikzpicture}
    \tikzstyle{vertex} = [circle,fill=black!10]
    \tikzstyle{selected vertex} = [vertex, fill=red!50]

    \tikzstyle{selected edge} = [draw,line width=1pt,-,red!100]
    \tikzstyle{edge} = [-,black,line width=1pt]
\end{tikzpicture}
```

Her fortelle vi at `vertex`er (eller noder), skal være sirkler. Markerte noder skal være fylt med rød farge.

Kanter skal tegnes som sorte linjer (`[-, black ...]`). Og markerte kanter skal være røde.



4.1 Tegne grafen

For å plassere nodene rundt om på arket sier man hvor man vil de skal være ved hjelp at koordinater.

```
\begin{tikzpicture}[scale=5]
    \tikzstyle{vertex}          = [circle,fill=black!10]
    \tikzstyle{selected vertex} = [vertex, fill=red!50]

    \tikzstyle{selected edge}   = [draw,line width=1pt,-,red!100]
    \tikzstyle{edge}            = [-,black,line width=1pt]

    \node[vertex]          (v1) at (1.25,1.7) {1};
    \node[vertex]          (v2) at (1.5,1.1)  {2};
    \node[selected vertex] (v3) at (0.9,1.5)  {3};
    \node[vertex]          (v4) at (1.6,1.5)  {4};
    \node[vertex]          (v5) at (1,1.1)    {5};

    \draw[edge]          (v1)--(v2)--(v3)--(v4)--(v5)--(v1);
    \draw[selected edge] (v1)--(v2);
\end{tikzpicture}
```

Nodene defineres ved å først bruke nøkkelordet `node`, så fortelle hvilken type node dette er. I dette tilfellet, så er det enten `vertex` eller `selected vertex` som vi har definert med `tikzstyle`. Nodens navn bruker man kun i egen kode, slik som når man skal tegne opp kantene. Koordinatene `(x,y)` forteller hvor vi vil plassere noden, og verdien er innholdet i noden.

```
\node[type of node] (node name) at (x,y) {value};
```

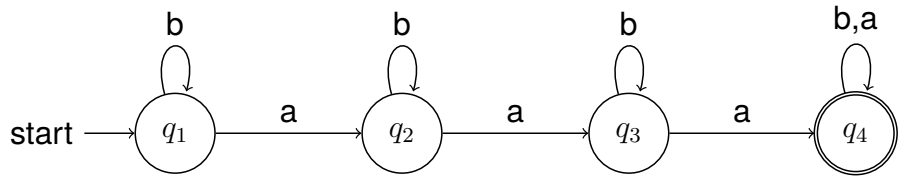
Kantene tegnes likt som linjer fra seksjon 1. Men her gir vi nøkkelordet `draw` en av to stiler, som vi definerte med `tikzstyle`. Enten `edge` eller `selected egde`.

```
\draw[type of edge] (from node) -- (to node);
```

5 Automater

Denne måten å tegne automater på krever at man inkluderer

```
\usetikzlibrary{automata}
```




```
\begin{tikzpicture}[->,auto,node distance=3cm,line width=0.2mm]
  \node[initial,state   (A)                  {$q_1$};
  \node[state]          (B) [right of=A]     {$q_2$};
  \node[state]          (C) [right of=B]     {$q_3$};
  \node[state,accepting](D) [right of=C]     {$q_4$};

  \path (A) edge [loop above] node           {b}   (A)
            edge node                        {a}   (B)
        (B) edge [loop above] node           {b}   (B)
            edge node                        {a}   (C)
        (C) edge [loop above] node           {b}   (C)
            edge node                        {a}   (D)
        (D) edge [loop above] node           {b,a} (D);
\end{tikzpicture}
```

For denne måten å tegne automater på, så settes alle parametre som beskriver automaten i definisjonen til `tikzpicture`. Her har automaten følgende egenskaper:

```
{tikzpicture}[->, auto, node distance=3cm, line width=0.2mm]
```

Dette forteller oss at automaten skal tegnes med piler (`->`), nodene skal ha avstand på 3 cm, og linjene en tykkelse på 0,2 mm. Auto stiller teksten *over* linjene, i stedet for *på* linjene.

5.1 Automatens tilstander

En automat har tre typer tilstander: starttilstanden, vanlig tilstand(er), og akepterende tilstand(er).

```
\node[state] (node name) {state name};
```

I tillegg til `[state]`, så kan man ha med opsjonen `[initial, state]` for starttilstanden, eller `[state, accepting]` for aksepterende tilstand.

5.2 Stien gjennom automaten

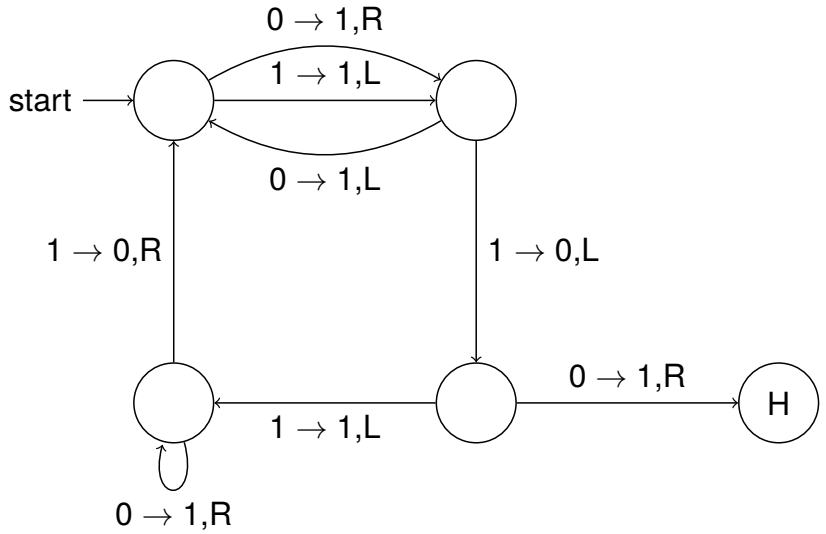
Stien tegnes gjennom en `path`. Denne konstrueres på følgende vis:

`\path (from state) edge [opt.] node {weight} (to state)`

Her kan `[opt]` være `loop above/below`, `bend left/right`.

Flittig bever

Her er en flittig 4-bever. Denne automaten dekker de fleste opsjoner.




```
\begin{tikzpicture}[->,auto,node distance=4cm,line width=0.2mm]
  \node[initial,state] (A)              {};
  \node[state]         (B) [below of=A] {};
  \node[state]         (C) [right of=A] {};
  \node[state]         (D) [below of=C] {};
  \node[state]         (E) [right of=D] {H};

  \path (A) edge node              {1 $\rightarrow$ 1,L}  (C)
        (A) edge [bend left] node  {0 $\rightarrow$ 1,R}  (C)
        (C) edge [bend left] node  {0 $\rightarrow$ 1,L}  (A)
        (B) edge node              {1 $\rightarrow$ 0,R} (A)
        (B) edge [loop below] node {0 $\rightarrow$ 1,R}  (B)
        (D) edge node              {1 $\rightarrow$ 1,L}  (B)
        (C) edge node              {1 $\rightarrow$ 0,L}  (D)
        (D) edge node              {0 $\rightarrow$ 1,R}  (E);
\end{tikzpicture}
```

6 Logiske porter

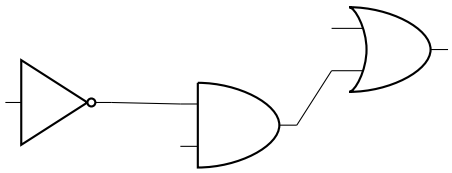
Noe som er kjekt å vite om er også logiske porter i Circuitikz. Dette får du ved å inkludere pakken:

```
\usepackage{circuitikz}
```

Siden dette *ikke* er TikZ jobber vi ikke i miljøet `tikzpicture`, men i miljøet `circuitikz`.

```
\begin{circuitikz} \draw
    <kode her>
\end{circuitikz}
```

6.1 Eksempel på en liten krets



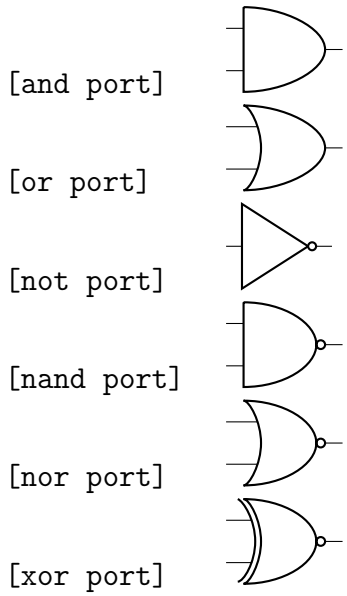
```
\begin{circuitikz} \draw
    (-3,0.3) node[not port] (not) {}
    (0,0)    node[and port] (and) {}
    (2,1)    node[or port]  (or)  {}

    (not.out) -- (and.in 1)
    (and.out) -- (or.in 2);
\end{circuitikz}
```

Det fungerer på samme måte som når vi tegner noder i TikZ. Vi starter med koordinatene, så definerer vi hva slags node (port) vi vil ha, og til slutt en evt. merkelapp.

```
(x,y) node [what kind of port] (name of port) {label}
```

Portens navn er valgfritt, og brukes kun i din egen kode.

6.2 Oversikt over forskjellige porter i Circuitikz



Figur 5: Forskjellige porter i Circuitikz

7 Ressurser

Gøyale eksempler

- Enderman
- Dartboard
- India map

Lære mer?

- TikZ & PGF Manual
- Introduksjon til Circuitikz
- Tankekart med TikZ
- Generere TikZ-kode fra GeoGebra

Visste du at..

Roger Antonsens bok «Logiske Metoder» er full av TikZ/PGF?