The \texttt{tqft} package: codebase

Andrew Stacey
loopspace@mathforge.org

June 1, 2017

1 Introduction

This is a package for drawing TQFT diagrams using PGF/TikZ. Its inspiration was a question and answer on the website \url{http://tex.stackexchange.com}.

2 Implementation

2.1 Old Version: Node Shapes

\begin{verbatim}
\RequirePackage{pgfkeys}
\RequirePackage{pgf}

We can view the cobordisms from the input or output ends, the implementation of the choice is to draw an arc from 0 to 180 or from 0 to -180 so we just need to track minus signs. These macros are for that.
\def\pgf@tqft@minus{-}
\let\pgf@tqft@upper\@empty
\let\pgf@tqft@lower\pgf@tqft@minus

Some helpful extra functions.
\tqftset is our equivalent of \texttt{tikzset}.
\def\tqftset#1{\pgfkeys{/pgf/tqft/.cd,#1}}
\end{verbatim}
\texttt{\textbackslash tqft@process} This macro applies our flow transformation to the given coordinates, with the result stored in \texttt{\pgf@x} and \texttt{\pgf@y}.

\begin{verbatim}
7 \def\tqft@process#1#2{\%
8 \edef\tqft@px{#1}
9 \edef\tqft@py{#2}
10 \pgf@process{
11 \pgftransformreset
12 \let\tikz@transform=\pgfutil@empty
13 \expandafter\tikzset\expandafter{\tqft@transformation}
14 \tikz@transform
15 \pgfpointtransformed{\pgfqpoint{\tqft@px}{\tqft@py}}}
16 } \%
17 }
\end{verbatim}

Declare some dimension registers to hold the specifications of the cobordism.

\begin{verbatim}
18 \newdimen\tqft@xa
19 \newdimen\tqft@xb
20 \newdimen\tqft@c
21 \newdimen\tqft@ch
22 \newdimen\tqft@h
23 \newdimen\tqft@s
24 \newdimen\tqft@w
25 \newif\iftqft@within@node
\end{verbatim}

Now we set up all the keys that we'll need in the course of this shape

\begin{verbatim}
26 \pgfkeys{
27 \tikz/tqft/use nodes/.is choice,
28 \tikz/tqft/use nodes/true/.code=\%
29 \tikzset{
30 tqft/.style=\%
31 \tikz/shape=tqft cobordism,
32 /pgf/tqft,
33 \tikz/every tqft/.try
34 },
35 tqft/.unknown/.code=\%
36 \let\tqft@searchname=\pgfkeyscurrentname\%
37 \pgfkeys{
38 \pgf/tqft/tqft@searchname={##1}\%
39 }
40 },
41 }
42 }
\end{verbatim}

If not using nodes, set the defaults for the library

\begin{verbatim}
43 \tikz/tqft/use nodes/false/.code=\%
\end{verbatim}

\begin{verbatim}
44 \tikz/tqft/use nodes/false/.code=\%
\end{verbatim}
\tikzset{
  tqft/.style={%
    pic type=cobordism,
    tqft/.cd,
    every tqft/.try,
  },
}

Pass unknown keys on to TikZ.
\tqft/.unknown/.code={%
  \let\tqft@searchname=\pgfkeyscurrentname%
  \pgfkeys{%
    /tikz/\tqft@searchname={##1}
  }
}\%

/\tikz/tqft/use nodes=true,

This deals with unknown keys, passing them on to TikZ.

/\pgf/tqft/.unknown/.code={%
  \let\tqft@searchname=\pgfkeyscurrentname%
  \pgfkeysalso{%
    /tikz/\tqft@searchname={#1}
  }
}\%

/\pgf/tqft/.is family,

Let’s play happy families!
/\pgf/tqft/.is family,
/\pgf/tqft,

This sets our shape to be the boundary circle
boundary circle/.style={
  /\tikz/shape=tqft boundary circle
},

These set our number of boundary components
incoming boundary components/.initial=5,
outgoing boundary components/.initial=4,

This is the “horizontal” offset of the first outgoing component from the first incoming one.
offset/.initial=0,

This is the “vertical” separation between boundary components.
cobordism height/.initial=2cm,

This is the “horizontal” separation between boundary components.
boundary separation/.initial=2cm,

These are the “horizontal” and “vertical” radii, respectively, of the boundary components (perhaps poorly named!).
circle width/.initial=10pt,
circle depth/.initial=5pt,
These control the separation between the node and its anchors.

```latex
outer xsep/.initial=0pt,
outer ysep/.initial=0pt,
onter sep/.style={
  outer xsep=#1,
onter ysep=#1
},
```

This is our flow control. The `flow` key installs a transformation to be applied to our node shape. The possible transformations are stored in the following keys. They aren’t just rotations so that the numbering is always “top to bottom” or “left to right”.

```latex
flow/.code={%
  \pgfkeys{/pgf/tqft/flow transformation/.expand twice/.expand once=\pgfkeysvalueof{/pgf/tqft/flow transformation #1}}%
},
flow transformation south/.initial={},
flow transformation north/.initial=%
  xscale=-1,rotate=180%
},
flow transformation east/.initial=%
  rotate=90,xscale=-1%
},
flow transformation west/.initial=%
  rotate=270%
},
flow transformation/.initial={},
```

These control the direction from which we view the cobordism.

```latex
view from/.is choice,
view from/incoming/.code={%
  \let\pgf@tqft@upper\pgf@tqft@minus
  \let\pgf@tqft@lower\@empty%
},
view from/outgoing/.code={%
  \let\pgf@tqft@lower\pgf@tqft@minus
  \let\pgf@tqft@upper\@empty%
},
```

The next set of keys are for styling the different pieces of a cobordism.

```latex
boundary lower style contents/.initial={},
boundary lower style/.code=%
  \pgfkeys{/pgf/tqft/boundary lower style contents/.style={%
    /tikz/.cd,#1%
  }}%
},
boundary style contents/.initial={},
boundary style/.code=%
  \pgfkeys{/pgf/tqft/boundary style contents/.style={%
    /tikz/.cd,#1%
  }}%
},
```
boundary upper style/.code={%
\pgfkeys{/pgf/tqft/boundary upper style contents/.style={%
\tikz/.cd,#1
%}
%}
},
cobordism style contents/.initial={},
cobordism style/.code={%
\pgfkeys{/pgf/tqft/cobordism style contents/.style={%
\tikz/.cd,#1%
%}
%}
},

The next set of keys define some default shapes.

pair of pants/.style={
/tikz/tqft,
incoming boundary components=1,
outgoing boundary components=2,
offset=-.5
%}
/tikz/tqft pair of pants/.style={
/pgf/tqft/pair of pants,
%}
reverse pair of pants/.style={
/tikz/tqft,
incoming boundary components=2,
outgoing boundary components=1,
offset=.5
%}
/tikz/tqft reverse pair of pants/.style={
/pgf/tqft/reverse pair of pants,
%}
cylinder to prior/.style={
/tikz/tqft,
incoming boundary components=1,
outgoing boundary components=1,
offset=-.5
%}
/tikz/tqft cylinder to prior/.style={
/pgf/tqft/cylinder to prior,
%}
cylinder to next/.style={
/tikz/tqft,
incoming boundary components=1,
outgoing boundary components=1,
This is a generic cobordism shape.

Save our specifications: incoming and outgoing boundary components

and the offset (in units of boundary components) between the leading incoming and outgoing components (regarded as a shift of the outgoing components relative to the incoming)

Now we save our dimensions: height, separation, the radii of the boundary circles, and outer seps, and the heights of the control points.
This is the internal transformation that is in place

\saveddimen{\tqft@control}{
  \pgfkeysgetvalue{/pgf/tqft/cobordism height}{\tqft@tempa}
  \pgfkeysgetvalue{/pgf/tqft/circle depth}{\tqft@tempb}
  \pgfmathsetlength{\pgf@x}{.5 * \tqft@tempa - 4 * \tqft@tempb}
}\savedmacro{\tqft@transformation}{%
  \pgfkeysgetvalue{/pgf/tqft/flow transformation}{\tqft@transformation}
}\savedmacro{\tqft@transformation}{%

For the externally available anchors, we need to save a few things as well.

Position of first incoming boundary in internal coordinates
\savedanchor{\tqft@start@incoming}{%
  \pgfmathsetlength{\pgf@x}{-(\max(\pgfkeysvalueof{/pgf/tqft/incoming boundary components} - \pgfkeysvalueof{/pgf/tqft/offset}) * \pgfkeysvalueof{/pgf/tqft/boundary separation} / 2)}
  \pgfmathsetlength{\pgf@y}{.5 * \pgfkeysvalueof{/pgf/tqft/cobordism height}}
}\savedanchor{\tqft@start@incoming}{%

Position of first outgoing boundary in internal coordinates
\savedanchor{\tqft@start@outgoing}{%
  \pgfmathsetlength{\pgf@x}{-(\max(\pgfkeysvalueof{/pgf/tqft/incoming boundary components} - 2 * \pgfkeysvalueof{/pgf/tqft/offset}) * \pgfkeysvalueof{/pgf/tqft/boundary separation} / 2)}
  \pgfmathsetlength{\pgf@y}{-.5 * \pgfkeysvalueof{/pgf/tqft/cobordism height}}
}\savedanchor{\tqft@start@outgoing}{%

For completeness, we record the size of the text box (not that we expect any text, but you never know)
\savedanchor{\tqft@textsize}{%
  \pgf@y=-.5\ht\pgfnodeparttextbox%
  \pgf@x=-.5\wd\pgfnodeparttextbox%
}\anchor{centre}{\pgfpointorigin}
\anchor{center}{\pgfpointorigin}
\anchor{text}{\tqft@textsize}
\anchor{north}{%
  \pgf@ya=\tqft@height\relax
  \pgf@yb=.5\pgf@ya
  \advance\pgf@yb by \tqft@outerysep\relax
  \tqft@process{0pt}{\the\pgf@yb}
}\anchor{north}{%
\anchor{south}{%
  \pgf@yb=\tqft@height\relax
  \pgf@ya=.5\pgf@yb
  \advance\pgf@ya by \tqft@outerysep\relax
  \tqft@process{0pt}{\the\pgf@ya}
}\anchor{south}{%
\anchor{west}{%
  \pgf@ya=\tqft@height\relax
  \pgf@yb=.5\pgf@ya
  \advance\pgf@yb by \tqft@outerysep\relax
  \tqft@process{0pt}{\the\pgf@yb}
}\anchor{west}{%
\anchor{centre}{\pgfpointorigin}
\anchor{center}{\pgfpointorigin}
\anchor{text}{
  \tqft@textsize
}\anchor{north}{%
  \pgf@ya=\tqft@height\relax
  \pgf@yb=.5\pgf@ya
  \advance\pgf@yb by \tqft@outerysep\relax
  \tqft@process{0pt}{\the\pgf@yb}
}\anchor{north}{%
\begingroup
\newcommand{\pgftx}{\pgf@x}
\newcommand{\pgfty}{\pgf@y}
\newcommand{\tqft@width}{\pgftx}
\newcommand{\tqft@start@outgoing}{\advance\pgftx by -\tqft@width}
\newcommand{\tqft@start@incoming}{\pgftx=\pgf@x}
\newcommand{\tqft@process}[2]{\the#1\the#2}
\newcommand{\tqft@outerxsep}{.5}
\newcommand{\tqft@outerysep}{.5}
\newcommand{\tqft@incoming}{.5}
\newcommand{\tqft@outgoing}{.5}
\newcommand{\tqft@separation}{.5}
\newcommand{\anchor}[1]{%}
\tqft@start@outgoing
\pgftx=\pgf@x
\advance\pgftx by -\tqft@width
\pgfty=\pgf@y
\tqft@process{\the\pgftx}{\the\pgfty}
\anchor{east}{%}
\tqft@start@incoming
\pgftx=\pgf@x
\pgfmathsetlength{\pgftx}{\pgftx + (\tqft@incoming - 1) * \tqft@separation}
\advance\pgftx by \tqft@width
\pgfty=\pgf@y
\tqft@process{\the\pgftx}{\the\pgfty}
\anchor{north west}{%}
\tqft@start@incoming
\pgftx=\pgf@x
\pgfty=\pgf@y
\pgftx=.5}\pgf@x
\pgfty=.5}\pgf@y
\advance\pgftx by -\tqft@width
\advance\pgfty by \tqft@outerysep
\advance\pgftx by -\tqft@outerxsep
\tqft@process{\the\pgftx}{\the\pgfty}
\anchor{south west}{%}
\tqft@start@outgoing
\pgftx=\pgf@x
\pgfty=\pgf@y
\pgftx=.5}\pgf@x
\pgfty=.5}\pgf@y
\advance\pgftx by -\tqft@width
\advance\pgfty by -\tqft@outerysep
\advance\pgftx by -\tqft@outerxsep
\tqft@process{\the\pgftx}{\the\pgfty}
\endgroup
To define anchors at the boundary components requires a bit of trickery borrowed from the "regular polygon" shape.
342 \advance\c@pgf@counta-1\relax%  
343 \repeatpgfmathloop%  
344}  
345\expandafter\pgfutil@addtomacro\csname pgf@sh@s@tqft cobordism\endcsname{  
346\c@pgf@counta \tqft@incoming\relax%  
347\advance\c@pgf@counta-1\relax  
348\pgfmathloop%  
349\ifnum\c@pgf@counta>0\relax%  
350\pgfutil@ifundefined{pgf@anchor@tqft cobordism@after incoming boundary:\space\the\c@pgf@counta}{  
351\expandafter\xdef\csname pgf@anchor@tqft cobordism@after incoming boundary:\space\the\c@pgf@counta\endcsname{  
352\tqft@start@incoming  
353\pgfmathsetlength\noexpand\pgf@y{.25 * \noexpand\pgf@y +.75 * \noexpand\tqft@control + \noexpand\tqft@outerysep}  
354\pgfmathsetlength\noexpand\pgf@x{.25 * \noexpand\pgf@x + \noexpand\tqft@separation}  
355\tqft@process\noexpand\the\noexpand\pgf@x\noexpand\pgf@y}  
356}  
357\c@pgf@counta0\relax%  
358\advance\c@pgf@counta-1\relax%  
359\repeatpgfmathloop%  
360}  
361\expandafter\pgfutil@addtomacro\csname pgf@sh@s@tqft cobordism\endcsname{  
362\c@pgf@counta \tqft@outgoing\relax%  
363\advance\c@pgf@counta-1\relax  
364\pgfmathloop%  
365\ifnum\c@pgf@counta>0\relax%  
366\pgfutil@ifundefined{pgf@anchor@tqft cobordism@after outgoing boundary:\space\the\c@pgf@counta}{  
367\expandafter\xdef\csname pgf@anchor@tqft cobordism@after outgoing boundary:\space\the\c@pgf@counta\endcsname{  
368\tqft@start@outgoing  
369\pgfmathsetlength\noexpand\pgf@y{.25 * \noexpand\pgf@y -.75 * \noexpand\tqft@control - \noexpand\tqft@outerysep}  
370\pgfmathsetlength\noexpand\pgf@x{.25 * \noexpand\pgf@x + \noexpand\tqft@separation}  
371\tqft@process\noexpand\the\noexpand\pgf@x\noexpand\pgf@y}  
372}  
373\c@pgf@counta0\relax%  
374\advance\c@pgf@counta-1\relax%  
375\repeatpgfmathloop%  
376}  

Now we define the background path. This is the upper part of the cobordism.

377 \backgroundpath{

Apply the internal transformation
378 \let\tikz@transform=\pgfutil@empty  
379 \expandafter\tikzset\expandafter{\tqft@transformation}  
380 \tikz@transform

Convert the boundary separation and width to lengths
381 \pgfmathsetlength\noexpand\tqft@width{\noexpand\tqft@width}  
382 \pgfmathsetlength\noexpand\tqft@sep{2*\noexpand\tqft@width}

Compute the starting position of the incoming boundary components so that we get the centre anchor on the centre of the cobordism
383 \tqft@start@incoming
Do we have any incoming boundary components at all?

\begin{verbatim}
\ifnum\tqft@incoming>0
Yes, so move to the position of the first and draw it
\pgfpathmoveto{\pgfqpoint{\tqft@xa}{\tqft@h}}
\pgfpatharc{\pgf@tqft@upper180}{0}{\tqft@width and \tqft@depth}
\end{verbatim}

Do we have any more incoming boundary components?

\begin{verbatim}
\ifnum\tqft@incoming>1
Yes, so iterate over the remaining incoming boundary components
\foreach \tqft@k in {2,...,\tqft@incoming} {
\advance\tqft@xa by \tqft@k\tqft@s
\advance\tqft@xb by \tqft@k\tqft@s
\advance\tqft@xb by -2\tqft@s
\advance\tqft@xa by -\tqft@s
\pgfpathcurveto{\pgfqpoint{\tqft@xb}{\tqft@c}}{\pgfqpoint{\tqft@xa}{\tqft@c}}{\pgfqpoint{\tqft@xa}{\tqft@h}}
\pgfpatharc{\pgf@tqft@upper180}{0}{\tqft@width and \tqft@depth}
}
\fi
\end{verbatim}

If we don’t have any outgoing boundary components, may as well close up now.

\begin{verbatim}
\ifnum\tqft@outgoing=0
\advance\tqft@xb by \tqft@incoming\tqft@s
\advance\tqft@xb by -\tqft@s
\pgfmathsetlength{\tqft@ch}{\min(0,\max(-\tqft@h,\tqft@h - (\tqft@h - \tqft@c) \times (abs(\tqft@xb - \tqft@xa) - \tqft@w)/\tqft@s + 1)))}
\pgfpathcurveto{\pgfqpoint{\tqft@xb}{\tqft@ch}}{\pgfqpoint{\tqft@xa}{-\tqft@ch}}{\pgfqpoint{\tqft@xa}{-\tqft@h}}
\fi
\fi
\end{verbatim}

Shift down to the outgoing components, if we have any

\begin{verbatim}
\ifnum\tqft@outgoing>0
\advance\tqft@xb by \tqft@incoming\tqft@s
\advance\tqft@xb by -\tqft@s
\pgfmathsetlength{\tqft@ch}{\min(0,\max(-\tqft@h,\tqft@h - (\tqft@h - \tqft@c) \times (abs(\tqft@xb - \tqft@xa) - \tqft@w)/\tqft@s + 1)))}
\pgfpathcurveto{\pgfqpoint{\tqft@xb}{\tqft@ch}}{\pgfqpoint{\tqft@xa}{-\tqft@ch}}{\pgfqpoint{\tqft@xa}{-\tqft@h}}
\else
\pgfpathmoveto{\pgfqpoint{\tqft@xa}{-\tqft@h}}
\fi
\tqft@xb=\tqft@xa
\advance\tqft@xb by -\tqft@w
\end{verbatim}
Now draw the lower components
\pgfpatharc{0}{\pgf@tqft@upper180}{\tqft@width and \tqft@depth}

Now iterate over the remaining outgoing boundary components
\ifnum\tqft@outgoing>1
\foreach \tqft@k in {2,...,\tqft@outgoing} { 
  \advance\tqft@xa by -\tqft@k\tqft@s
  \advance\tqft@xb by -\tqft@k\tqft@s
  \advance\tqft@xb by 2\tqft@s
  \advance\tqft@xa by \tqft@s
  \pgfpathcurveto{\pgfqpoint{\tqft@xb}{-\tqft@c}}{\pgfqpoint{\tqft@xa}{-\tqft@c}}{\pgfqpoint{\tqft@xa}{-\tqft@h}}
  \pgfpatharc{0}{\pgf@tqft@upper180}{\tqft@width and \tqft@depth}
}\fi

Shift back up to the incoming components, if we had any, otherwise arc back to our starting point
\ifnum\tqft@incoming>0
  \pgfmathsetlength{\tqft@xa}{-(\max(\tqft@incoming - 1,\tqft@outgoing - 1 + \tqft@offset) + \tqft@incoming - 1 + \tqft@offset) * \tqft@separation/2}
  \pgfmathsetlength{\tqft@h}{0.5 * \tqft@height}
\else
  \pgfmathsetlength{\tqft@xa}{-(\max(\tqft@incoming - 1,\tqft@outgoing - 1 + \tqft@offset) + \tqft@incoming - 1 + \tqft@offset) * \tqft@separation/2}
  \pgfmathsetlength{\tqft@h}{0.5 * \tqft@height}
\fi

Close the path
\pgfpathclose

End of background path Now we define the behind background path. This is the lower part of the boundary circles.
\behindbackgroundpath{
Apply the internal transformation
\let\tikz@transform=\pgfutil@empty
\expandafter\tikzset\expandafter{\tqft@transformation}
\tikz@transform
Convert the boundary separation and width to lengths
\pgfmathsetlength{\tqft@separation}{\tqft@separation}
\pgfmathsetlength{\tqft@w}{2*\tqft@width}

Compute the starting position of the incoming boundary components so that we get the centre anchor on the centre of the cobordism
\pgfmathsetlength{\tqft@xa}{-(\max(\tqft@incoming - 1,\tqft@outgoing - 1 + \tqft@offset) + \tqft@incoming - 1 + \tqft@offset) * \tqft@separation/2}
\pgfmathsetlength{\tqft@h}{0.5 * \tqft@height}

This section draws the boundary circles

Initialise the TikZ path settings and read in the style options for the boundary

\tikz@mode@fillfalse\%
\tikz@mode@drawfalse\%
\let\tikz@mode=\pgfutil@empty
\let\tikz@options=\pgfutil@empty
\tqftset{boundary style contents}
\tikz@mode
\tikz@options

Do we have any incoming boundary components at all?
\ifnum\tqft@incoming>0
Yes, so iterate over them
\foreach \tqft@k in {1,...,\tqft@incoming} {
\advance\tqft@xa by \tqft@k\tqft@s
\pgfpathellipse{\pgfqpoint{\tqft@xa}{\tqft@h}}{\pgfqpoint{\tqft@width}{0pt}}{\pgfqpoint{0pt}{\tqft@depth}}
}\fi

Now iterate over the outgoing boundary components, if we have any
\ifnum\tqft@outgoing>0
\pgfmathsetlength{\tqft@xa}{\tqft@xa + (\tqft@outgoing + \tqft@offset + 1) * \tqft@separation}
\foreach \tqft@k in {1,...,\tqft@outgoing} {
\advance\tqft@xa by -\tqft@k\tqft@s
% \advance\tqft@xa by \tqft@s
\pgfpathellipse{\pgfqpoint{\tqft@xa}{-\tqft@h}}{\pgfqpoint{\tqft@width}{0pt}}{\pgfqpoint{0pt}{\tqft@depth}}
}\fi
\edef\tikz@temp{\noexpand\pgfusepath{\iftikz@mode@fill fill,\fi\iftikz@mode@draw draw\fi}}%
\tikz@temp
}

This section draws the lower parts of the boundary circles
{
Initialise the TikZ path settings and read in the style options for the boundary
\tikz@mode@fillfalse\%
\tikz@mode@drawfalse\%
\let\tikz@mode=\pgfutil@empty
\let\tikz@options=\pgfutil@empty
\tqftset{boundary lower style contents}
\tikz@mode
\tikz@options
\advance\tqft@xa by .5\tqft@w

Do we have any incoming boundary components at all?
\ifnum\tqft@incoming>0
Yes, so iterate over them
\foreach \tqft@k in {1,...,\tqft@incoming} {
    \advance\tqft@xa by \tqft@k\tqft@s
    \pgfpathmoveto{\pgfqpoint{\tqft@xa}{\tqft@h}}
    \pgfpatharc{0}{\pgf@tqft@lower180}{\tqft@width and \tqft@depth}
}\fi

Now iterate over the outgoing boundary components, if we have any
\ifnum\tqft@outgoing>0
    \pgfmathsetlength{\tqft@xa}{\tqft@xa + (\tqft@outgoing + \tqft@offset + 1) * \tqft@separation}
    \foreach \tqft@k in {1,...,\tqft@outgoing} {
        \advance\tqft@xa by -\tqft@k\tqft@s
        \pgfpathmoveto{\pgfqpoint{\tqft@xa}{-\tqft@h}}
        \pgfpatharc{0}{\pgf@tqft@lower180}{\tqft@width and \tqft@depth}
    }\fi
\edef\tikz@temp{\noexpand\pgfusepath{\iftikz@mode@fill fill,\fi\iftikz@mode@draw draw\fi}}\	ikz@temp
}\}

End of behind background path.
Now we define the before background path. This is the upper part of the boundary circles and the cobordism edge.
\beforebackgroundpath{
    We don't apply the internal transformation as it is already in place from the \backgroundpath. Convert the boundary separation and width to lengths
    \pgfmathsetlength{\tqft@s}{\tqft@separation}
    \pgfmathsetlength{\tqft@w}{2*\tqft@width}

    Compute the starting position of the incoming boundary components so that we get the centre anchor on the centre of the cobordism
    \pgfmathsetlength{\tqft@xa}{-(max(\tqft@incoming - 1,\tqft@outgoing - 1 + \tqft@offset) + \tqft@offset) * \tqft@s/2 - \tqft@width}
    \tqft@xb=\tqft@xa
    \advance\tqft@xb by \tqft@w
    \tqft@c=\tqft@control\relax
    \pgfmathsetlength{\tqft@h}{.5 * \tqft@height}

    This section draws the non-boundary part of the cobordism.
    {\tikz@mode@fillfalse\tikz@mode@drawfalse\let\tikz@mode=\pgfutil@empty

    Initialise the TikZ path settings and read in the style options for the boundary
    \tikz@mode@fillfalse\tikz@mode@drawfalse
    \let\tikz@mode=\pgfutil@empty

14
\let\tikz@options=\pgfutil@empty
\tqftset{cobordism style contents}
\tikz@mode
\tikz@options
\% Do we have any incoming boundary components at all?
\% \begin{macrocode}
\ifnum\tqft@incoming>0
Do we have more than one?
\ifnum\tqft@incoming>1
Yes, so iterate over the remaining incoming boundary components
\foreach \tqft@k in {2,...,\tqft@incoming} {
\advance\tqft@xa by \tqft@k\tqft@s
\advance\tqft@xb by \tqft@k\tqft@s
\advance\tqft@xb by -2\tqft@s
\advance\tqft@xa by -\tqft@s
\pgfpathmoveto{\pgfqpoint{\tqft@xb}{\tqft@h}}
\pgfpathcurveto{\pgfqpoint{\tqft@xb}{\tqft@c}}{\pgfqpoint{\tqft@xa}{\tqft@c}}{\pgfqpoint{\tqft@xa}{\tqft@h}}
}\fi
\fi
If we don’t have any outgoing boundary components, may as well close up now.
\ifnum\tqft@outgoing=0
\advance\tqft@xb by \tqft@incoming\tqft@s
\advance\tqft@xb by -\tqft@s
\pgfmathsetlength{\tqft@ch}{min(0,max(-\tqft@h,\tqft@h - (\tqft@h - \tqft@c) * ((abs(\tqft@xb - \tqft@xa) - \tqft@w)/\tqft@s + 1)))}
\pgfpathmoveto{\pgfqpoint{\tqft@xb}{\tqft@h}}
\pgfpathcurveto{\pgfqpoint{\tqft@xb}{\tqft@ch}}{\pgfqpoint{\tqft@xa}{-\tqft@ch}}{\pgfqpoint{\tqft@xa}{-\tqft@h}}
\fi
\fi
Shift down to the outgoing components, if we have any
\ifnum\tqft@outgoing>0
\advance\tqft@xb by \tqft@incoming\tqft@s
\advance\tqft@xb by -\tqft@s
\pgfmathsetlength{\tqft@xa}{\tqft@xa + (\tqft@outgoing - 1 + \tqft@offset) * \tqft@separation + 2*\tqft@width}
If we had incoming boundaries, this is a curveto, otherwise it’s a moveto
\ifnum\tqft@incoming>0
\pgfmathsetlength{\tqft@ch}{min(0,max(-\tqft@h,\tqft@h - (\tqft@h - \tqft@c) * ((abs(\tqft@xb - \tqft@xa) - \tqft@w)/\tqft@s + 1)))}
\pgfpathmoveto{\pgfqpoint{\tqft@xb}{\tqft@h}}
\pgfpathcurveto{\pgfqpoint{\tqft@xb}{\tqft@ch}}{\pgfqpoint{\tqft@xa}{-\tqft@ch}}{\pgfqpoint{\tqft@xa}{-\tqft@h}}
\else
\pgfpathmoveto{\pgfqpoint{\tqft@xa}{-\tqft@h}}
\fi
\tqft@xb=\tqft@xa
\advance\tqft@xb by -\tqft@w
Now draw the lower components
\pgfpathmoveto{\pgfqpoint{\tqft@xb}{-\tqft@h}}

15
Now iterate over the remaining outgoing boundary components
\begin{verbatim}
\iftqft@outgoing>1
\foreach \tqft@k in \{2,\ldots,\tqft@outgoing\} {
\advance\tqft@xa by -\tqft@k\tqft@s
\advance\tqft@xb by -\tqft@k\tqft@s
\advance\tqft@xb by 2\tqft@s
\advance\tqft@xa by \tqft@s
\pgfpathcurveto{\pgfqpoint{\tqft@xb}{-\tqft@c}}{\pgfqpoint{\tqft@xa}{-\tqft@c}}{\pgfqpoint{\tqft@xa}{-\tqft@h}}
\advance\tqft@xa by -\tqft@w
\pgfpathmoveto{\pgfqpoint{\tqft@xa}{-\tqft@h}}
}\fi
\end{verbatim}

Shift back up to the incoming components, if we had any, otherwise arc back to our starting point
\begin{verbatim}
\iftqft@incoming>0
\pgfmathsetlength{\tqft@xa}{\tqft@xa - (\tqft@outgoing -1 + \tqft@offset) * \tqft@separation - 2*\tqft@width}
\pgfmathsetlength{\tqft@ch}{\tqft@h - (\tqft@h - \tqft@c) * (abs(\tqft@xb - \tqft@xa) - \tqft@w)/\tqft@s + 1)}
\pgfpathcurveto{\pgfqpoint{\tqft@xb}{-\tqft@ch}}{\pgfqpoint{\tqft@xa}{\tqft@ch}}{\pgfqpoint{\tqft@xa}{\tqft@h}}
\else
\pgfmathsetlength{\tqft@ch}{\tqft@h - (\tqft@h - \tqft@c) * (abs(\tqft@xb - \tqft@xa) - \tqft@w)/\tqft@s + 1)}
\pgfpathcurveto{\pgfqpoint{\tqft@xb}{-\tqft@ch}}{\pgfqpoint{\tqft@xa}{-\tqft@ch}}{\pgfqpoint{\tqft@xa}{-\tqft@h}}
\fi
\end{verbatim}

This section draws the upper parts of the boundary circles
\begin{verbatim}
\edef\tikz@temp{\noexpand\pgfusepath{\iftikz@mode@fill fill,\fi% \iftikz@mode@draw draw\fi%}}%
\tikz@temp
\end{verbatim}

This section draws the upper parts of the boundary circles
\begin{verbatim}
\let\tqft@bdry@path=\pgfutil@empty
\let\tqft@bdry@node@path=\pgfutil@empty
\pgfsyssoftpath@setcurrentpath{\tqft@bdry@path}{\pgfutil@empty}
\tikz@mode@fillfalse%
\tikz@mode@drawfalse%
\let\tikz@mode=\pgfutil@empty
\let\tikz@options=\pgfutil@empty
\tqftset{boundary upper style contents}
\tikz@mode
\tikz@options
\advance\tqft@xa by -\tqft@s
\advance\tqft@xa by \tqft@s
\end{verbatim}

Do we have any incoming boundary components at all?
\ifnum\tqft@incoming>0

Yes, so iterate over them

\foreach \tqft@k in {1,...,\tqft@incoming} {
\advance\tqft@xa by \tqft@k\tqft@s
\pgfpathmoveto{\pgfqpoint{\tqft@xa}{\tqft@h}}
\pgfpatharc{0}{\pgf@tqft@upper180}{\tqft@width and \tqft@depth}
\ifx\tikz@fig@name\pgfutil@empty
  \else
  \{\advance\tqft@xa by \tqft@width
  \pgftransformshift{\pgfqpoint{\tqft@xa}{\tqft@h}}
  \tqft@within@nodetrue
  \pgfsyssoftpath@getcurrentpath{\tqft@bdry@path}
  \pgfsyssoftpath@setcurrentpath{\tqft@bdry@node@path}
  \pgfnode{tqft boundary circle}{centre}{}{\tikz@fig@name space incoming \tqft@k}{}
  \pgfsyssoftpath@getcurrentpath{\tqft@bdry@node@path}
  \pgfsyssoftpath@setcurrentpath{\tqft@bdry@path}
\}
\fi
}

\fi

Now iterate over the outgoing boundary components, if we have any

\ifnum\tqft@outgoing>0
\pgfmathsetlength{\tqft@xa}{\tqft@xa + (\tqft@outgoing + \tqft@offset + 1) * \tqft@separation}
\foreach \tqft@k in {1,...,\tqft@outgoing} {
\advance\tqft@xa by \tqft@k\tqft@s
% \advance\tqft@xa by \tqft@s
\pgfpathmoveto{\pgfqpoint{\tqft@xa}{-\tqft@h}}
\pgfpatharc{0}{\pgf@tqft@upper180}{\tqft@width and \tqft@depth}
\ifx\tikz@fig@name\pgfutil@empty
  \else
  \{
  \pgfmathtruncatemacro{\tqft@l}{\tqft@outgoing + 1 - \tqft@k}
  \advance\tqft@xa by \tqft@width
  \pgftransformshift{\pgfqpoint{\tqft@xa}{-\tqft@h}}
  \tqft@within@nodetrue
  \pgfsyssoftpath@getcurrentpath{\tqft@bdry@path}
  \pgfsyssoftpath@setcurrentpath{\tqft@bdry@node@path}
  \pgfnode{tqft boundary circle}{centre}{}{\tikz@fig@name space outgoing \tqft@l}{}
  \pgfsyssoftpath@getcurrentpath{\tqft@bdry@node@path}
  \pgfsyssoftpath@setcurrentpath{\tqft@bdry@path}
\}
\fi
\fi

\edef\tikz@temp{\noexpand\pgfusepath{\iftikz@mode@fill fill,\fi\iftikz@mode@draw draw\fi%}
boundary circle shape This is a the shape of the boundary circles

```
\pgfdeclare SHAPE {tqft boundary circle} {
Now we save our dimensions: height, separation, and the radii of the boundary
circles

\savedimen{\tqft@height}{\pgf@x=\pgfkeysvalueof{/pgf/tqft/cobordism height}}
\savedimen{\tqft@separation}{\pgf@x=\pgfkeysvalueof{/pgf/tqft/boundary separation}}
\savedimen{\tqft@width}{\pgf@x=\pgfkeysvalueof{/pgf/tqft/circle width}}
\savedimen{\tqft@depth}{\pgf@x=\pgfkeysvalueof{/pgf/tqft/circle depth}}

For the externally available anchors, we need to save the declared transformation; we save the actual transformation, not the macro that points to it. If we’re
called within the main cobordism shape, the transformation is already applied so
we ignore it.

\savedimacro{\tqft@transformation}{%\iftqft@within@node
\let\tqft@transformation=\pgfutil@empty\else
\pgfkeysgetvalue{/pgf/tqft/flow transformation}{\tqft@transformation}\fi
}  
\savedanchor{\tqft@centre}{\pgfpointorigin}

For completeness, we record the size of the text box (not that we expect any text,
but you never know)

\savedanchor{\tqft@textsize}{\pgf@y=-.5\ht\pgfnodeparttextbox\pgf@x=-.5\wd\pgfnodeparttextbox}

These are our externally available anchors

\anchor{centre}{\tqft@centre} \anchor{center}{\tqft@centre} \anchor{text}{\tqft@textsize} \anchor{next}{\tqft@process{\tqft@separation}{0pt}} \anchor{prior}{\tqft@process{-\tqft@separation}{0pt}} \anchor{above}{\tqft@process{\tqft@depth}{0pt}} \anchor{next}{\tqft@process{\tqft@height}{0pt}}
```
The anchor border is the ellipse, but we need to take into account the possible transformation. (This isn’t right if the origin is shifted.) At the moment, ‘0 degrees’ is interpreted in the transformed coordinate system. Should provide a system whereby that can be interpreted in the main coordinate system.

This next `\pgf@process` makes the angles absolute. Comment it out to make the angles relative.

Now we define the background path. This is the upper part of the cobordism.

Apply the internal transformation if we’re not within a node

Draw the boundary circle

We draw the upper and lower arcs again with the appropriate styles
2.2 New Version: Picture Shapes

Issue a warning if the pic syntax is not available.
\ifcsname pgfk@/handlers/.pic/.@cmd\endcsname
\else
\pgfwarning{This library only works with TikZ 3.0 or later; for earlier versions of TikZ use the TQFT package}
\fi

For the boundaries, we need elliptical node shapes.
\usetikzlibrary{shapes.geometric}

We can view the cobordisms from the input or output ends, the implementation of the choice is to draw an arc from 0 to 180 or from 0 to -180 so we just need to track minus signs. These macros are for that.
\def\pgf@tqft@minus{-}
\let\pgf@tqft@upper@empty
\let\pgf@tqft@lower\pgf@tqft@minus

Split an anchoring coordinate. The $y$-value is simply multiplied by the cobordism height (but pointing downwards, so that 1 is level with the outgoing boundary). The $x$-value is multiplied by the boundary separation, but is shifted so that at the incoming boundary level, or above, then it is in line with the incoming boundaries and similarly at the outgoing boundary level, or below, it is in line with the outgoing boundaries.
\def\tqft@split(#1,#2){%
\pgfmathsetmacro\tqft@y{#2 * (-\tqft@val{cobordism height})}%
\pgfmathsetmacro\tqft@x{(#1 - 1 + max(min(#2,1),0)*\tqft@val{offset}) * \tqft@val{boundary separation}}%
\let\tqft@shift{(\tqft@x pt, \tqft@y pt)}%
Now we set up all the keys that we'll need in the course of this shape.

\tikzset{
    Fix for the fact that the alias key doesn't use the prefix and suffix.
    \tikzset{
        pic alias/.code={%
            \tikz@fig@mustbenamed
            \expandafter\def\expandafter\tikz@alias\expandafter{\tikz@alias\pgfnodealias{\tikz@pp@name{#1}}{\tikz@fig@name}}%
        },
    }

This key is our basic installer key, setting the pic and putting us in the right key family.

\tikzset{
    tqft/.style={%
        pic type=cobordism,
        every tqft/.try,
        tqft/.cd,
    },
}

This deals with unknown keys, passing them on to TikZ.

\tikzset{
    tqft/.unknown/.code={%
        \tikzf@searchname=\pgfkeyscurrentname%
        \pgfkeysalso{%
            /tikz/tqft\tikzf@searchname={#1}
        }
    },
}

Let's play happy families!

\tikzset{
    tqft/.cd,
}

These set our number of boundary components and genus.

\tikzset{
    incoming boundary components/.initial=5,
    outgoing boundary components/.initial=4,
    skip incoming boundary components/.initial={},
    skip outgoing boundary components/.initial={},
    genus/.initial = 0,
}

This is the “horizontal” offset of the first outgoing component from the first incoming one.

\tikzset{
    offset/.initial=0,
}

This is the “vertical” separation between boundary components.

\tikzset{
    cobordism height/.initial=2cm,
}

This is the “horizontal” separation between boundary components.

\tikzset{
    boundary separation/.initial=2cm,
}

These are the “horizontal” and “vertical” radii, respectively, of the boundary components.

\tikzset{
    circle x radius/.initial=10pt,
    circle y radius/.initial=5pt,
}

These control the direction from which we view the cobordism.

\tikzset{
    view from/.is choice,
    view from/incoming/.code=\%
We simulate node placement using the following key.

\begin{verbatim}
784 \let\pgf@tqft@upper\pgf@tqft@minus
785 \let\pgf@tqft@lower\@empty
786 },
787 view from/outgoing/.code=%
788 \let\pgf@tqft@lower\pgf@tqft@minus
789 \let\pgf@tqft@upper\@empty
790 },
\end{verbatim}

We simulate node placement using the following key.

\begin{verbatim}
791 anchor/.initial = none,
\end{verbatim}

The next set of keys define some default shapes.

\begin{verbatim}
792 pair of pants/.style={
793 /tikz/tqft,
794 incoming boundary components=1,
795 outgoing boundary components=2,
796 offset=-.5
797 },
798 /tikz/tqft pair of pants/.style={
799 /tikz/tqft/pair of pants,
800 },
801 reverse pair of pants/.style={
802 /tikz/tqft,
803 incoming boundary components=2,
804 outgoing boundary components=1,
805 offset=.5
806 },
807 /tikz/tqft reverse pair of pants/.style={
808 /tikz/tqft/reverse pair of pants,
809 },
810 cylinder to prior/.style={
811 /tikz/tqft,
812 incoming boundary components=1,
813 outgoing boundary components=1,
814 offset=-.5
815 },
816 /tikz/tqft cylinder to prior/.style={
817 /tikz/tqft/cylinder to prior,
818 },
819 cylinder to next/.style={
820 /tikz/tqft,
821 incoming boundary components=1,
822 outgoing boundary components=1,
823 offset=.5
824 },
825 /tikz/tqft cylinder to next/.style={
826 /tikz/tqft/cylinder to next,
827 },
828 cylinder/.style={
829 /tikz/tqft,
830 incoming boundary components=1,
\end{verbatim}
outgoing boundary components=1 },
/tikz/tqft cylinder/.style={
/tikz/tqft/cylinder,
},
cup/.style={
/tikz/tqft,
incoming boundary components=1,
outgoing boundary components=0 },
/tikz/tqft cup/.style={
/tikz/tqft/cup,
},
cap/.style={
/tikz/tqft,
incoming boundary components=0,
outgoing boundary components=1 },
/tikz/tqft cap/.style={
/tikz/tqft/cap,
},
}

This is a little helper macro for getting the values of tqft keys.
\def\tqft@val#1{\pgfkeysvalueof{/tikz/tqft/#1}}

Now we define the code for the actual cobordism shape.
\tikzset{
cobordism/.pic={

Defining the cobordism paths. This holds the full boundary path of the cobordism shape.
\gdef\tqft@fullpath{}

This is a list of the edge pieces without the boundary circles.
\global\let\tqft@blist\pgfutil@gobble%
This punches the holes (if there are any) in the cobordism shape.
\global\let\tqft@gclip\pgfutil@gobble%
This is a list of the paths for drawing the holes.
\global\let\tqft@glist\pgfutil@gobble%
This collects any coordinates that are to be defined (it appears to be difficult
to define them as we go along).
\global\let\tqft@clist\pgfutil@gobble%
This collects any coordinates that can be used to shift the shape that aren’t to be
defined using \tqft@clist.
\global\let\tqft@alist\pgfutil@gobble
These will be lists of the boundary components, divided into sets as to whether or not they are rendered. For the outgoing ones, we need too lists because they are rendered in the opposite order to how they are labelled:

\begin{verbatim}
\global\let\tqft@ibdrylist=\pgfutil@gobble
\global\let\tqft@cibdrylist=\pgfutil@gobble
\global\let\tqft@obdrylist=\pgfutil@gobble
\global\let\tqft@cobdrylist=\pgfutil@gobble
\global\let\tqft@robdrylist=\pgfutil@gobble
\global\let\tqft@rcobdrylist=\pgfutil@gobble
\end{verbatim}

The first stage is to iterate over the incoming boundary components (if there are any), building up the various paths.

\begin{verbatim}
\ifnum\tqft@val{incoming boundary components}>0\relax
\begin{verbatim}
 We have some so draw the half circle for the first component. Note that we use \pgf@tqft@upper to flip the sign of the start angle depending on the view from setting.
\end{verbatim}
\begin{verbatim}
\edef\tqft@temp{\noexpand\pgfutil@in@{,\k,}{,\tqft@val{skip incoming boundary components},}}
\tqft@temp
\ifpgfutil@in@
\xdef\tqft@cibdrylist{\tqft@cibdrylist,\k}
\else
\xdef\tqft@ibdrylist{\tqft@ibdrylist,\k}
\fi
\end{verbatim}
\end{verbatim}
\end{verbatim}

Add the curve and the arc.

\begin{verbatim}
\ifnum\tqft@val{incoming boundary components}>1\relax
\begin{verbatim}
\foreach \k in {2,...,\tqft@val{incoming boundary components}} {
\begin{verbatim}
\pgfmathsetmacro\xpos{\tqft@val{circle x radius}}
\end{verbatim}
\begin{verbatim}
\pgfmathsetmacro\cpos{\xpos + \tqft@val{circle x radius}}
\end{verbatim}
\end{verbatim}
\end{verbatim}
\end{verbatim}
\end{verbatim}

Add the curve and the arc.

\begin{verbatim}
\foreach \k [evaluate=\xpos using (\k-1)*\tqft@val{boundary separation} -\tqft@val{circle x radius}, \cpos using (\xpos + \tqft@val{boundary separation} + \tqft@val{circle x radius} + \tqft@val{circle y radius})/2] in \tqft@ibdrylist {
\begin{verbatim}
\pgfmathsetmacro\xpos{\tqft@val{circle x radius}}
\end{verbatim}
\end{verbatim}
\end{verbatim}

Add the curve and the arc.
But for the edge path, just add the curve to the list.
\def\tqft@blist{%
  \tqft@blist,incoming boundary \k/incoming/{% (\xppos pt,0) .. controls +(0,-\tqft@val{cobordism height}/3) and +(0,-\tqft@val{cobordism height}/3) .. (%}
}\% We add a coordinate at the midpoint of the curve.
\def\tqft@clist{%
  \tqft@clist,-between incoming \kmo space and \k/{(\cpos pt,-\tqft@val{cobordism height}/4)}% \}
}\%
And add the centre to the list for available shifts.
\def\tqft@alist{%
  \tqft@alist,-incoming boundary \k/{(\kmo * \tqft@val{boundary separation},0)}% \}
\%}
\fi

We're at the edge of the last incoming boundary component. What we do now depends on whether or not there are outgoing boundary components.
\ifnum\tqft@val{outgoing boundary components}>0\relax
There are, so we add a curve from the end of the last incoming to the last outgoing component to the full path,
\pgfmathsetmacro\xppos{(\tqft@val{outgoing boundary components} -1+\tqft@val{offset})% \pgfmathsetmacro\tqft@ht{\abs{\tqft@val{incoming boundary components} - \tqft@val{outgoing boundary components}} - \tqft@val{outgoing boundary components}}% \pgfmathsetmacro\tqft@ht{1/3 + 2/3*\tqft@ht/(\tqft@ht + 1)}% \def\tqft@fullpath{%
  \tqft@fullpath .. controls +(0,-\tqft@ht*\tqft@val{cobordism height}) and +(0,\tqft@ht*\tqft@val{cobordism height})% and the edge path.
\def\tqft@blist{%
  \tqft@blist,between last incoming and last outgoing/incoming and outgoing/{% (\tqft@val{incoming boundary components} + \tqft@val{boundary separation}) + \tqft@val{outgoing boundary components}}% \}
\%}
In addition, we add a coordinate at the midpoint.
\pgfmathsetmacro\xppos{((\xppos + (\tqft@val{incoming boundary components} -1) * \tqft@val{boundary separation}) + \tqft@val{outgoing boundary components})/2}% \% 
\else
There aren't any outgoing boundary components so we loop back to the start. We adjust the height of the control points to take into account the overall width.
\def\tqft@fullpath{\%
  \tqft@fullpath
  \node at (0,0) {
  \tqft@ht} \tqft@val{cobordism height} \tqft@val{cobordism height} \}
\def\tqft@blist{\%
  \tqft@blist,between first incoming and last incoming/incoming and outgoing/{% 
  \node at (0,0) {
  \tqft@val{incoming boundary components} \tqft@val{boundary separation} + 
  \tqft@val{circle x radius} \tqft@val{boundary separation},0pt 
  \node at (0,0) {
  \tqft@ht} \tqft@val{cobordism height} \tqft@val{cobordism height} \}
\def\tqft@clist{\%
  \tqft@clist,-between first incoming and last incoming/{\xppos pt,-\tqft@ht*3/4} 
\fi
\else
There weren’t any incoming boundary components, so we test to see if there were any outgoing ones and move to the start of them. 
\begin{itemize}
  \item \ifnum\tqft@val{outgoing boundary components}>0\relax
  \pgfmathsetmacro\xppos{\tqft@val{outgoing boundary components} -1+\tqft@val{offset}}
  \pgfmathsetmacro\xppos{\xppos pt,-\tqft@ht*\tqft@val{cobordism height}}
  \node at (0,0) {
  \tqft@ht} \tqft@val{cobordism height} \tqft@val{cobordism height} \}
\fi
\fi
\end{itemize}
\def\tqft@fullpath{\%
  \tqft@fullpath
  \arc[end angle=pgf@tqft@upper180, start angle=0, x radius=\tqft@val{circle x radius}, \tqft@val{circle y radius}]
\fi
\fi
\def\tqft@alist{\%
  \tqft@alist,-outgoing boundary \tqft@val{outgoing boundary components}/{(\xppos pt + \tqft@val{circle x radius},-\tqft@ht*3/4)},-outgoing boundary/{(\tqft@val{offset} \tqft@val{boundary separation},-\tqft@ht*3/4)} 
\fi
\else
We’re done with the incoming boundary components, now we’re set up for the outgoing ones. However we got there, if we have outgoing boundary components then we’re now located at the start of them, although we’re counting backwards.
\begin{itemize}
  \item \ifnum\tqft@val{outgoing boundary components}>0\relax
  \pgfmathsetmacro\xppos{\tqft@val{outgoing boundary components} -1+\tqft@val{offset}}
  \pgfmathsetmacro\xppos{\xppos pt,-\tqft@ht*\tqft@val{cobordism height}}
  \node at (0,0) {
  \tqft@ht} \tqft@val{cobordism height} \tqft@val{cobordism height} \}
\fi
\fi
\end{itemize}
\def\tqft@fullpath{\%
  \tqft@fullpath
  \arc[end angle=pgf@tqft@upper180, start angle=0, x radius=\tqft@val{circle x radius}, \tqft@val{circle y radius}]
\fi
\fi
\def\tqft@alist{\%
  \tqft@alist,-outgoing boundary \tqft@val{outgoing boundary components}/{(\xppos pt + \tqft@val{circle x radius},-\tqft@ht*3/4)},-outgoing boundary/{(\tqft@val{offset} \tqft@val{boundary separation},-\tqft@ht*3/4)} 
\fi
\else
Do we have more than one boundary component? 
\begin{itemize}
  \item \ifnum\tqft@val{outgoing boundary components}>1\relax
\fi
\fi
\end{itemize}
Yes, so add a curve and arc for each.

\foreach \k [evaluate=\k as \ok using int(\tqft@val{outgoing boundary components} - \k + 1)] in {2,...,\tqft@val{outgoing boundary components}} {
  \edef\tqft@temp{\noexpand\pgfutil@in@{,\ok,}{,\tqft@val{skip outgoing boundary components}},}
  \tqft@temp
  \ifpgfutil@in@
    \xdef\tqft@cobdrylist{\tqft@cobdrylist,\k}
  \else
    \xdef\tqft@cobdrylist{\tqft@cobdrylist,\k}
  \fi
}
\ifx\tqft@cobdrylist\pgfutil@gobble
  \else
    \foreach \k [remember=\k as \kmo (initially 1), evaluate=\k as \xpos using (\tqft@val{outgoing boundary components} - \k + \tqft@val{offset})*\tqft@val{boundary separation} + \tqft@val{circle x radius},] in \tqft@cobdrylist {
      \pgfmathsetmacro\xppos{(\tqft@val{outgoing boundary components} - \kmo + \tqft@val{offset})*\tqft@val{boundary separation} - \tqft@val{circle x radius}}
      \pgfmathsetmacro\cpos{((\xpos + \xppos)/2)}
      \pgfmathsetmacro\nk{int(\tqft@val{outgoing boundary components} - \k + 1)}
      \pgfmathsetmacro\nkpo{int(\tqft@val{outgoing boundary components} - \kmo + 1)}

      Both are added to the full path.
      \edef\tqft@fullpath{\tqft@fullpath
        \pgfpathmoveto{(\cpos pt,-\tqft@val{cobordism height})}
        \pgfpatharc{angle=0}{angle=\pgf@tqft@upper180}{x radius=\tqft@val{circle x radius} y radius=\tqft@val{circle y radius}}
        \pgfpathclose
      }
    }
  \fi
\fi

Now we're at the end of the outgoing boundary components (well, the start actually). What we do now depends on whether or not there are any incoming boundary components.
\ifnum\tqft@val{incoming boundary components}>0\relax

Just the arc for the edge paths.
\foreach \k [remember=\k as \kmo (initially 1), evaluate=\k as \xpos using (\tqft@val{outgoing boundary components} - \kmo + \tqft@val{offset})*\tqft@val{boundary separation} + \tqft@val{circle x radius},] in \tqft@cobdrylist {
  \pgfmathsetmacro\xppos{(\tqft@val{outgoing boundary components} - \kmo + \tqft@val{offset})*\tqft@val{boundary separation} - \tqft@val{circle x radius}}
  \pgfmathsetmacro\cpos{((\xpos + \xppos)/2)}
  \pgfmathsetmacro\nk{int(\tqft@val{outgoing boundary components} - \k + 1)}
  \pgfmathsetmacro\nkpo{int(\tqft@val{outgoing boundary components} - \kmo + 1)}

  And a coordinate at the midpoint.
  \edef\tqft@clist{\tqft@clist,-between outgoing \nk\space and \nkpo/outgoing/{(\cpos pt,-3*\tqft@val{cobordism height})}]
}

And add the centre to the list for available shifts.
\foreach \k [remember=\k as \kmo (initially 1), evaluate=\k as \xpos using (\tqft@val{outgoing boundary components} - \kmo + \tqft@val{offset})*\tqft@val{boundary separation} + \tqft@val{circle x radius},] in \tqft@cobdrylist {
  \pgfmathsetmacro\xppos{(\tqft@val{outgoing boundary components} - \kmo + \tqft@val{offset})*\tqft@val{boundary separation} - \tqft@val{circle x radius}}
  \pgfmathsetmacro\cpos{((\xpos + \xppos)/2)}
  \pgfmathsetmacro\nk{int(\tqft@val{outgoing boundary components} - \k + 1)}
  \pgfmathsetmacro\nkpo{int(\tqft@val{outgoing boundary components} - \kmo + 1)}

  And add the centre to the list for available shifts.
  \edef\tqft@alist{\tqft@alist,-outgoing boundary \nk/{{(\xpos pt - \tqft@val{circle x radius} ,-\tqft@val{circle y radius})}]
}
\fi
\fi
There are, so we draw the path back up.
\begin{verbatim}
\pgfmathsetmacro{\tqft@ht}{1/3 + 2/3*abs(\tqft@val{offset})/(abs(\tqft@val{offset}) + 1)}
\xdef{\tqft@fullpath}{\tqft@fullpath .. controls +(0,\tqft@ht*\tqft@val{cobordism height}) and +(0,-\tqft@ht*\tqft@val{cobordism height})}
\end{verbatim}
And the edge path does the same.
\begin{verbatim}
\xdef{\tqft@blist}{\tqft@blist,between first incoming and first outgoing/incoming and outgoing/{\tqft@val{offset} * \tqft@val{boundary separation} - \tqft@val{circle x radius},\tqft@val{circle x radius}} .. controls +(0,\tqft@ht*\tqft@val{cobordism height}) and +(0,-\tqft@ht*\tqft@val{cobordism height})}
\end{verbatim}
Add a coordinate at the midpoint.
\begin{verbatim}
\xdef{\tqft@clist}{\tqft@clist,-between first incoming and first outgoing/{(\tqft@val{offset} * \tqft@val{boundary separation}/2 - \tqft@val{circle x radius},-\tqft@val{cobordism height}/2)}
\end{verbatim}
No incoming boundary components so loop back to the other end of the outgoing boundary components.
\begin{verbatim}
\pgfmathsetmacro{\xppos}{(\tqft@val{outgoing boundary components} - 1 + \tqft@val{offset}) * \tqft@val{boundary separation} + \tqft@val{circle x radius}}
\pgfmathsetmacro{\tqft@ht}{1/3 + 2/3*(\tqft@val{outgoing boundary components} - 1)/\tqft@val{outgoing boundary components}}
\xdef{\tqft@fullpath}{\tqft@fullpath .. controls +(0,\tqft@ht*\tqft@val{cobordism height}) and +(0,\tqft@ht*\tqft@val{cobordism height}) .. (\xppos pt,-\tqft@val{cobordism height})
\end{verbatim}
Full path.
\begin{verbatim}
\xdef{\tqft@blist}{\tqft@blist,between first and last outgoing/incoming and outgoing/{(\tqft@val{offset} * \tqft@val{boundary separation} - \tqft@val{circle x radius},-\tqft@val{cobordism height}) .. controls +(0,\tqft@ht*\tqft@val{cobordism height}) and +(0,\tqft@ht*\tqft@val{cobordism height})}
\end{verbatim}
Add a coordinate at the midpoint.
\begin{verbatim}
\pgfmathsetmacro{\xppos}{(\tqft@val{outgoing boundary components}/2 + \tqft@val{offset} - 1/2) * \tqft@val{boundary separation}}
\pgfmathsetmacro{\tqft@ht}{1 - \tqft@ht*3/4}
\xdef{\tqft@clist}{\tqft@clist,-between first and last outgoing/{(\xppos pt,-\tqft@ht*\tqft@val{cobordism height})}}
\end{verbatim}
Now we define the clip path for the genus holes. We start with a big rectangle that ought to be big enough to contain the whole shape. We start with the top left corner.
\begin{verbatim}
\pgfmathsetmacro{\xpos}{(28}
\end{verbatim}
\input{tqft}

\begin{verbatim}
\tqft@val{outgoing boundary components} > 0 ?
\tqft@val{incoming boundary components} > 0 ?
\min(0,\tqft@val{offset}) : \tqft@val{offset}
\) : 0
\)
\*\tqft@val{boundary separation} - 2*\tqft@val{circle x radius}}%
\xdef\tqft@gclip{\xpos pt,2*\tqft@val{circle y radius}) rectangle }

Now the bottom right.
\pgfmathsetmacro\xpos{%
\tqft@val{outgoing boundary components} > 0 ?
\tqft@val{incoming boundary components} > 0 ?
\max(\tqft@val{incoming boundary components},\tqft@val{outgoing boundary components} + 
\) : \tqft@val{incoming boundary components}
\)-1)
\*\tqft@val{boundary separation} + 2*\tqft@val{circle x radius}}%
Together, these make a rectangle.
\xdef\tqft@gclip{\tqft@gclip (\xpos pt,-\tqft@val{cobordism height} - 2*\tqft@val{circle y radius})}%

Are there any holes?
\ifnum\tqft@val{genus}>0\relax
Yes, so first we need to figure out where to place them. We work out the left-hand
edge of the cobordism.
\pgfmathsetmacro\xpos{%
\tqft@val{outgoing boundary components} > 0 ?
\tqft@val{incoming boundary components} > 0 ?
\tqft@val{offset}/2 : \tqft@val{offset}
\) : 0
\)
\*\tqft@val{boundary separation} - \tqft@val{circle x radius}}%
Work out the height that the holes should be punched at.
\pgfmathsetmacro\ypos{%
\tqft@val{outgoing boundary components} > 0 ?
\tqft@val{incoming boundary components} > 0 ?
\tqft@val{offset}/2 : \tqft@val{offset}
\) : 0
\)
\-\tqft@val{cobordism height}/2 : -1 + \tqft@val{cobordism height}/3
\) : - \tqft@val{cobordism height}/3
\)\%
Start our clip path at this point
\xdef\tqft@gclip{%
\tqft@gclip (\xpos pt,\ypos pt)
}%
\end{verbatim}
Now work out the width of the cobordism, in units of circle half-widths. This may not be very accurate if there aren’t any boundary components of a given type.

\[
\pgfmathsetmacro\gsize{\begin{cases}
\tqft@val{outgoing boundary components} > 0 ? \\
( \tqft@val{incoming boundary components} > 0 ? \frac{\tqft@val{incoming boundary components} + \tqft@val{outgoing boundary components}}{2} : \tqft@val{outgoing boundary components} \\
-1) \times \tqft@val{boundary separation}/\tqft@val{circle x radius} + 2
\end{cases}}
\]

Each hole should take up three half-widths, but we want a little extra on the edges so the total number of half-widths we want is \(3g + 1\). Do we need to scale down the holes (we never scale up)? If so, \(gscale\) holds the overall scale factor and \(gxscale\) and \(gyscale\) are the resulting horizontal and vertical measurements.

The baseline is the size of the boundary circles.

\[
\pgfmathsetmacro\gscale{\min(1,\gsize/(3\tqft@val{genus}+1))}
\pgfmathsetmacro\gyscale{\tqft@val{circle y radius} \times \gscale \times .707}
\pgfmathsetmacro\gxscale{\tqft@val{circle x radius} \times \gscale}
\]

Each hole should take up 2 half widths, modulo scaling, so the total width used by the holes is \(2gs\) leaving \(w - 2gs\) left for the gaps which is divided in to \(g + 1\) lots.

\[
\pgfmathsetmacro\gsep{\frac{\gsize - 2\tqft@val{genus} \times \gscale}{\tqft@val{genus} + 1} \times \tqft@val{circle x radius}}
\]

We shift in by half of one unit of excess separation.

\[
\edef\tqft@gclip{\tqft@gclip + (\gsep/2 pt,0)}
\]

Some useful quantities.

\[
\pgfmathsetmacro\omrstwo{1 - 1/\sqrt{2}}
\pgfmathsetmacro\sqrtwo{\sqrt{2}}
\]

Now we iterate over the holes.

\[
\foreach\[evaluate=\k as \kmo using int(2 \times \k-1)] \k in \{1,...,\tqft@val{genus}\} {
\]

For the clipping path, we just want the bare hole.

\[
\edef\tqft@gclip{\tqft@gclip + (\gsep/2 pt,0) \times \omrstwo*(gyscale pt,0)}
\]

Move in by half an excess separation unit and move to the left-hand extent of the hole.

\[
++(\gsep/2 pt \times \omrstwo\times gxscale pt,0)
\]

Now curve up over the hole,

\[
.. controls +(\sqrtwo\\times\sqrtwo/3 pt,4/3\times gyscale pt) and +(-\sqrtwo\times\sqrtwo/3 pt,4/3\times gyscale pt)
.. ++(\sqrtwo\times\sqrtwo/3 pt,0)
\]

30
and return on the underside.

Lastly, move to the right-hand edge of the space taken up by this hole.

For the genus path we want to add the little “tails” which means that the two curves are different, and we need to take into account the view from direction.

Move to the starting point of the smaller curve and add that.

Add the larger of the two curves.

Add a coordinate at the centre of the hole.

Now we start to lay out the cobordism. Were we given a shift? If so, shift.
At each incoming boundary component we place an elliptical node of the right size.

\begin{verbatim}
\ifnum\tqft@val{incoming boundary components}>0\relax
  \ifx\tqft@ibdrylist\pgfutil@gobble
    \xdef\tqft@ibdrylist{1}
  \else
    \xdef\tqft@ibdrylist{1,\tqft@ibdrylist}
  \fi
  \foreach[evaluate=\k as \xpos using (\k-1)*\tqft@val{boundary separation}] \k in \tqft@ibdrylist {
    \node[node contents={}, ellipse, inner sep=0pt, outer sep=0pt, minimum width=2*\tqft@val{circle x radius}, minimum height=2*\tqft@val{circle y radius}, at={(\xpos pt,0)}, name=-incoming boundary \k, /tikz/tqft/every boundary component/.try, /tikz/tqft/every incoming boundary component/.try, /tikz/tqft/incoming boundary component \k/.try];
  }
\fi
\end{verbatim}

Add an alias for the first.

\begin{verbatim}
\path node also[pic alias=-incoming boundary] (-incoming boundary 1);\fi
\end{verbatim}

Same for the outgoing boundary components.

\begin{verbatim}
\ifnum\tqft@val{outgoing boundary components}>0\relax
  \ifx\tqft@obdrylist\pgfutil@gobble
    \xdef\tqft@obdrylist{1}
  \else
    \xdef\tqft@obdrylist{1,\tqft@obdrylist}
  \fi
  \foreach[evaluate=\k as \xpos using (\k-1)*\tqft@val{boundary separation}] \k in \tqft@obdrylist {
    \node[node contents={}, ellipse, inner sep=0pt, outer sep=0pt, minimum width=2*\tqft@val{circle x radius}, minimum height=2*\tqft@val{circle y radius}, at={(\xpos pt,0)}, name=-incoming boundary \k, /tikz/tqft/every boundary component/.try, /tikz/tqft/every skipped boundary component/.try, /tikz/tqft/skipped incoming boundary component \k/.try];
  }
\fi
\end{verbatim}

Add an alias for the first.

\begin{verbatim}
\path node also[pic alias=-incoming boundary] (-incoming boundary 1);\fi
\end{verbatim}

Same for the outgoing boundary components.
\begin{tikzpicture}[scale=0.7]
\tqft@outgoing boundary components\n\tqft@obdrylist\n\tqft@robdrylist\n\tqft@rcobdrylist\n\tqft@val{boundary separation}\n\tqft@val{circle x radius}\n\tqft@val{circle y radius}\n\tqft@val{cobordism height}\n\tqft@val{offset}\n\node (-outgoing boundary 1) [pic alias=-outgoing boundary] at (\xpos pt,-\tqft@val{cobordism height}) \tikz\path node also [pic alias=-outgoing boundary] (-outgoing boundary 1);\fi

Add an alias for the first.
\path node also [pic alias=-outgoing boundary] (-outgoing boundary 1);\fi
Now we draw the lower paths of the incoming boundary components.

\begin{verbatim}
\ifnum\tqft@val{incoming boundary components}>0\relax
\foreach[evaluate=\k as \xpos using (\k-1)*\tqft@val{boundary separation}] \k in \tqft@ibdrylist {
  \path[/tikz/tqft/every lower boundary component/.try,
    /tikz/tqft/every incoming lower boundary component/.try,
    /tikz/tqft/incoming lower boundary component \k/.try
  ] (\xpos pt - \tqft@val{circle x radius},0) arc[start angle=\pgf@tqft@lower180,end angle=0, x radius=\tqft@val{circle x radius}, y radius=\tqft@val{circle y radius}];
}\fi

Same for the outgoing boundary components.

\begin{verbatim}
\ifnum\tqft@val{outgoing boundary components}>0\relax
\foreach[evaluate=\k as \xpos using (\k-1+\tqft@val{offset})*\tqft@val{boundary separation}] \k in \tqft@robdrylist {
  \path[/tikz/tqft/every lower boundary component/.try,
    /tikz/tqft/every outgoing lower boundary component/.try,
    /tikz/tqft/outgoing lower boundary component \k/.try
  ] (\xpos pt - \tqft@val{circle x radius},-\tqft@val{cobordism height}) arc[start angle=\pgf@tqft@lower180,end angle=0, x radius=\tqft@val{circle x radius}, y radius=\tqft@val{circle y radius}];
}\fi
\end{verbatim}

Full outer path, clipped against the genus holes in case it is filled.

\begin{verbatim}
\begin{scope}
  \path[overlay,clip] \tqft@gclip;
  \path[/tikz/tqft/cobordism/.try,
    pic actions,
    /tikz/tqft/cobordism outer path/.try,
  ] \tqft@fullpath;
\end{scope}
\end{verbatim}

Now we draw the genus path, outside the clip. We view this as part of the full cobordism path so try to apply the same style as for the full path, but if that is filled then we turn the fill off. It can be turned back on again using the styles cobordism edge or genus style. We also apply the cobordism edge style as it could be thought of as part of the non-boundary edge. Finally, it has its own style to enable overrides if the other two get confused.

\begin{verbatim}
\begin{verbatim}
\begin{scope}\ifx\tqft@glist\pgfutil@gobble\else\foreach \tqft@gstyle/\tqft@gside/\tqft@gpath in \tqft@glist {\path[/tikz/tqft/cobordism/.try,
  pic actions,
  fill=none,
  shade=none,
  /tikz/tqft/cobordism edge/.try,
}\fi\end{scope}
\end{verbatim}
\end{verbatim}
\end{verbatim}

34
Now we redraw the non-boundary paths.

\ifx\tqft@blist\pgfutil@gobble
\else
\foreach \tqft@bstyle/\tqft@btype/\tqft@bpath in \tqft@blist {
\path [\tikz/tqft/cobordism edge/\tqft@bstyle/.try,
\tikz/tqft/cobordism outer edge/.try,
\tikz/tqft/between \tqft@btype/.try,
\tikz/tqft/\tqft@bstyle/.try]
\tqft@bpath;
}\fi

There were various coordinates that we wanted to define but couldn’t. Here, we put those in place.

\ifx\tqft@clist\pgfutil@gobble
\else
\foreach \name/\coord in \tqft@clist {
\path \coord node[coordinate,node contents={},name=\name];
}\fi

The last task is to draw the upper paths of the boundary components. First, incoming.

\ifnum\tqft@val{incoming boundary components}>0\relax
\foreach[evaluate=\k as \xpos using (\k-1)*\tqft@val{boundary separation}] \k in \tqft@ibdrylist {
\path [\tikz/tqft/every upper boundary component/.try,
\tikz/tqft/every incoming upper boundary component/.try,
\tikz/tqft/incoming upper boundary component \k/.try
] (\xpos pt - \tqft@val{circle x radius},0) arc[\pgf@tqft@upper\pgfmath@angle@at@meters,\pgf@tqft@upper\pgfmath@angle@at@meters,\tqft@val{circle x radius},\tqft@val{circle y radius}];
}\fi

Next, outgoing.

\ifnum\tqft@val{outgoing boundary components}>0\relax
\foreach[evaluate=\k as \xpos using (\k-1+\tqft@val{offset})*\tqft@val{boundary separation}] \k in \tqft@robdrylist {
\path [\tikz/tqft/every upper boundary component/.try,
\tikz/tqft/every outgoing upper boundary component/.try,
\tikz/tqft/outgoing upper boundary component \k/.try
] (\xpos pt - \tqft@val{circle x radius},-\tqft@val{cobordism height}) arc[\pgf@tqft@upper\pgfmath@angle@at@meters,\pgf@tqft@upper\pgfmath@angle@at@meters,-\tqft@val{circle x radius},\tqft@val{circle y radius}];
}\fi

35
We're done! Phew.