

# the Java program for 2-knot invariants

Glen Lancaster      Richard Larson      Jacob Towber

September 23, 2004

In Section 1 we briefly discuss some of the `Java` classes which are used; in Section 2 we describe how to compile and run the `Java` program `ProgramBt`; in Section 3 we discuss possible paths for future development.

## 1 The basic Java classes

In this section we give a brief overview of the more important of the classes which we define.

### Event

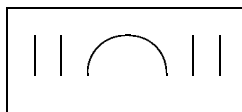
An `Event` is one of the following five possibilities: `Cup`, `Cap`, `NE`, `NW`, or `Null`.

In terms of [1], the event `NW` is a positive crossing  $X$ , and `NE` is a negative crossing  $\bar{X}$ .

### FEvent

A `Framed Event` is an `Event` with threads to the left and right of the `Event`. The `String` constructor for the class `FEvent` takes a parameter of the form  $[m, event, n]$ , where  $m$  is the number of threads to the left of  $event$ , and  $n$  is the number of threads to the right of  $event$ .

For example, `FEvent("[2, cap, 2]")` is



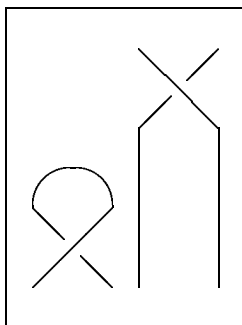
The `Event` is one of `Cap`, `Cup`, `NE`, `NW`, and `Null`. Note that  $event$  is case-independent.

To simplify testing for equality, the framed `Null` event with  $m$  threads to the left and  $n$  threads to the right is always represented internally with  $m + n$  threads to the left and 0 threads to the right, independent of the argument for the constructor.

## Still

A Still consists of a sequence of Framed Events. Pictorially, the framed events occur from the bottom to the top of the picture, as we read a string given in [1] from right to left. The String constructor for the class `Still` takes a list of framed events, separated by optional semicolons (;). The string is read from left-to-right, as opposed to the right-to-left convention in [1].

For example, `Still("[0,ne,2];[0,cap,2];[0,nw,0]")` is the Still



which would be given in [1] as  $X_{0,0} \cap_{0,2} \bar{X}_{0,2}$ .

Two Stills are equal if the sequences of Framed Events are equal (method: `equals(Still)`); they are approximately equal (method: `approxEquals(Still)`) if they are equal after removing all null Framed Events.

## ESI

An Elementary Still Interaction is a transition from one Still to another. These occur as 87 subclasses of the class `ESI`.

An ESI is specified by giving the starting (origin) and ending (terminus) Stills.

The tests to determine the origin and the terminus use “approximately equal” rather than “equal.”

We give in Appendix B a table of ESIs by name, origin, and terminus.

## Framed Elementary String Interaction

A Framed Elementary String Interaction is an ESI, preceded and followed by threads.

A FESI is specified by giving two Stills, which, when the same number of threads is removed from before and after each, specify an ESI.

## Flicker

A Flicker is a transition from one Still to another. It is specified by giving the two Stills, separated by `->`. The first Still has a pair of `S`, and the second Still

has a pair of F. The S and F are case-insensitive. The Still between the pair of S in the origin and the Still between the pair of F in the terminus must specify the origin and terminus of a FESI. The Still before the first S in the origin must be approximately equal to the Still before the first F in the terminus. The Still after the second S in the origin must be approximately equal to the Still after the second F in the terminus.

If the Still of the origin or terminus of the ESI is only a null framed event, the Ss or Fs may be adjacent.

Here is an example of a Flicker:

```
[2,cup,0]s[1,nw,1];[1,ne,1]s[2,cap,0];[0,cap,0];[0,cup,0];[0,cap,0]
->
[2,cup,0]ff[2,cap,0];[0,cap,0];[0,cup,0];[0,cap,0]
```

## Movie

A Movie is a sequence of Flickers —  $n$  Flickers are specified by giving  $n + 1$  Stills, each pair separated by `->`. Each Still has two S and two F, except for the first, which has only two S, and for the last, which has only two F.

The specification of a Movie is insensitive to the presence or absence of blanks and newlines within the Movie. The line following the Movie must begin with a period (`.`).

Appendix A contains an example of a file which contains two Movies.

## 2 Compiling and running ProgramBt

ProgramBt computes the amplitudes of each Movie in a file of Movies, both the amplitudes which are not necessarily preserved by the Movie Moves 31 of [1], and the amplitudes which are preserved.

ProgramBt writes a file usable in L<sup>A</sup>T<sub>E</sub>X as well as a log file, and writes its results to the standard output.

After the file `distrib.zip` is downloaded, unZIPped, and the Java source is compiled using the command

```
javac ProgramBt.java
```

you can run ProgramBt by using the command

```
java ProgramBt fn
```

where the file `fn.mov` contains a sequence of Movies. The contents of the file `fn.mov` is case insensitive, and may contain white space (embedded blanks and blank lines). Each Movie is terminated with a line which begins with a period (`.`).

ProgramBt writes its result to the standard output, to `fn.log`, and also in L<sup>A</sup>T<sub>E</sub>X format to `fn.tex`.

Lines in the input file which start with `#!` are copied to `fn.log`. Lines which start with `#` (but not with `#!`) are copied to the standard output, to `fn.log`, and to `fn.tex`. Each Movie is written to `fn.log`, with an alternate format using the ESI names given in Appendix B. The resulting invariants are written to the standard output, to `fn.log`, and to `fn.tex` (what is written to `fn.tex` has the `#` removed and is slightly reformatted).

A sample run of `ProgramBt` is given in Appendix A, together with its output files.

### 3 Future lines for development

With the exception of certain parameters related to computing amplitudes, the action of the programs is controlled by input files rather than by “hard-wired” code. This provides greater flexibility.

In order to investigate  $U$ -regular amplitudes for various  $U$ , we intend to make the parameters used to compute the amplitudes also set by input files. This will give total flexibility in investigating the amplitudes associated with the different  $U \subseteq \mathfrak{sl}$ .

## Appendix A

Here is a sample run of program `ProgramBt`:

```
[D:\knots]java ProgramBt knots0
# a simple 2-unknot
Amplitudes without Movie Moves 31:
  the 0th invariant is: 1*q^(-2)+1*q^(2)
  the 1th invariant is: 1*q^(-2)+1*q^(2)
  the 2th invariant is: 1*q^(-2)+1*q^(2)
  the 3th invariant is: 1*q^(0)
  the 4th invariant is: 1*q^(0)
Amplitudes with Movie Moves 31:
  the 0th invariant is: 1*q^(0)
  the 1th invariant is: 1*q^(0)
# simple torus
Amplitudes without Movie Moves 31:
  the 0th invariant is: 1*q^(-4)+2*q^(0)+1*q^(4)
  the 1th invariant is: 1*q^(-4)+2*q^(0)+1*q^(4)
  the 2th invariant is: 1*q^(-4)+2*q^(0)+1*q^(4)
  the 3th invariant is: 1*q^(0)
  the 4th invariant is: 1*q^(0)
Amplitudes with Movie Moves 31:
  the 0th invariant is: 1*q^(0)
  the 1th invariant is: 1*q^(0)
```

Here is the file knots0.mov, which contains two simple knotted surfaces: a sphere and a torus.

```
# a simple 2-unknot
ss
->
sf[0,cup,0];[0,cap,0]sf
->
ff
.
# simple torus
ss ->
f[0,cup,0]ss[0,cap,0]f ->
[0,cup,0]fs[0,cap,0][0,cup,0]fs[0,cap,0] ->
s[0,cup,0]ff[0,cap,0]s ->
ff
.
```

Here is the resulting log file from executing ProgramBt with this file:

```
# a simple 2-unknot
Movie:
^0<ESI6R>0^:
  ^0<ESI6>0^

Amplitudes without Movie Moves 31:
  the 0th invariant is:  $1*q^{(-2)}+1*q^{(2)}$ 
  the 1th invariant is:  $1*q^{(-2)}+1*q^{(2)}$ 
  the 2th invariant is:  $1*q^{(-2)}+1*q^{(2)}$ 
  the 3th invariant is:  $1*q^{(0)}$ 
  the 4th invariant is:  $1*q^{(0)}$ 
Amplitudes with Movie Moves 31:
  the 0th invariant is:  $1*q^{(0)}$ 
  the 1th invariant is:  $1*q^{(0)}$ 
# simple torus
Movie:
^0<ESI6R>0^:
  [0,Cup,0]^0<ESI7R>0^[0,Cap,0]:
  [0,Cup,0]^0<ESI7>0^[0,Cap,0]:
  ^0<ESI6>0^

Amplitudes without Movie Moves 31:
  the 0th invariant is:  $1*q^{(-4)}+2*q^{(0)}+1*q^{(4)}$ 
  the 1th invariant is:  $1*q^{(-4)}+2*q^{(0)}+1*q^{(4)}$ 
  the 2th invariant is:  $1*q^{(-4)}+2*q^{(0)}+1*q^{(4)}$ 
  the 3th invariant is:  $1*q^{(0)}$ 
  the 4th invariant is:  $1*q^{(0)}$ 
```

Amplitudes with Movie Moves 31:  
 the 0th invariant is:  $1*q^{(0)}$   
 the 1th invariant is:  $1*q^{(0)}$

ProgramB results in a similar log file.

Here is the resulting L<sup>A</sup>T<sub>E</sub>X file from executing ProgramBt with this file:

```
{\bf a simple 2-unknot:}
```

Amplitudes without Movie Moves 31:

```
\begin{eqnarray*}
\langle K \rangle_{0} & = & \& +q^{-2}+q^{2} \\
\langle K \rangle_{1} & = & \& +q^{-2}+q^{2} \\
\langle K \rangle_{2} & = & \& +q^{-2}+q^{2} \\
\langle K \rangle_{3} & = & \& +1 \\
\langle K \rangle_{4} & = & \& +1
\end{eqnarray*}
```

Amplitudes with Movie Moves 31:

```
\begin{eqnarray*}
\langle K \rangle_{0} & = & \& +1 \\
\langle K \rangle_{1} & = & \& +1
\end{eqnarray*}
{\bf simple torus:}
```

Amplitudes without Movie Moves 31:

```
\begin{eqnarray*}
\langle K \rangle_{0} & = & \& +q^{-4}+2+q^{4} \\
\langle K \rangle_{1} & = & \& +q^{-4}+2+q^{4} \\
\langle K \rangle_{2} & = & \& +q^{-4}+2+q^{4} \\
\langle K \rangle_{3} & = & \& +1 \\
\langle K \rangle_{4} & = & \& +1
\end{eqnarray*}
```

Amplitudes with Movie Moves 31:

```
\begin{eqnarray*}
\langle K \rangle_{0} & = & \& +1 \\
\langle K \rangle_{1} & = & \& +1
\end{eqnarray*}
```

## Appendix B

In this appendix we list the ESIs by name and give their origins and terminuses.

The ESI0 are provided to allow inputs that are not designed to work with programs which do not relax tests for equality of Stills to tests for approximate equality.

The ESI8 are provided to avoid difficulties with trivial moves which cannot be described with the other ESIs. For example,

```
[0,cup,0][0,cap,0]s[0,cup,0][2,cup,0]s[2,cap,0][0,cap,0]
->
[0,cup,0][0,cap,0]f[0,cup,0]s[0,cup,2]f[2,cap,0]s[0,cap,0]
->
[0,cup,0]s[0,cap,0][0,cup,0]sf[0,cap,0][0,cup,0]f[0,cap,0]
->
[0,cup,0]f[2,cup,0]s[0,cap,2]f[0,cap,0]s[0,cup,0][0,cap,0]
->
[0,cup,0][2,cup,0]f[2,cap,0][0,cap,0]f[0,cup,0][0,cap,0]
```

describes moving a circle positioned to the right of another circle down to the right of a third circle below the original circle.

Name	Origin	Terminus
ESI0cap	[0,Cap,0]	[2,Null,0];[0,Cap,0]
ESI0capR	[2,Null,0];[0,Cap,0]	[0,Cap,0]
ESI0cup	[0,Cup,0]	[0,Cup,0];[2,Null,0]
ESI0cupR	[0,Cup,0];[2,Null,0]	[0,Cup,0]
ESI0nedn	[0,NE,0]	[0,NE,0];[2,Null,0]
ESI0nednR	[0,NE,0];[2,Null,0]	[0,NE,0]
ESI0neup	[0,NE,0]	[2,Null,0];[0,NE,0]
ESI0neupR	[2,Null,0];[0,NE,0]	[0,NE,0]
ESI0null	[0,Null,0]	[0,Null,0]
ESI0wn dn	[0,NW,0]	[0,NW,0];[2,Null,0]
ESI0wn dnR	[0,NW,0];[2,Null,0]	[0,NW,0]
ESI0nwup	[0,NW,0]	[2,Null,0];[0,NW,0]
ESI0nwupR	[2,Null,0];[0,NW,0]	[0,NW,0]
ESI1	[0,Cup,0];[0,NE,0]	[0,Cup,0]
ESI1R	[0,Cup,0]	[0,Cup,0];[0,NE,0]
ESI1tau	[0,Cup,0];[0,NW,0]	[0,Cup,0]
ESI1Rtau	[0,Cup,0]	[0,Cup,0];[0,NW,0]
ESI1ttau	[0,NE,0];[0,Cap,0]	[0,Cap,0]
ESI1Rttau	[0,Cap,0]	[0,NE,0];[0,Cap,0]
ESI1t	[0,NW,0];[0,Cap,0]	[0,Cap,0]
ESI1Rt	[0,Cap,0]	[0,NW,0];[0,Cap,0]
ESI2	[0,NE,0];[0,NW,0]	[2,Null,0]
ESI2R	[2,Null,0]	[0,NE,0];[0,NW,0]
ESI2tau	[0,NW,0];[0,NE,0]	[2,Null,0]
ESI2Rtau	[2,Null,0]	[0,NW,0];[0,NE,0]
ESI3bmt	[0,NE,1];[1,NE,0];[0,NE,1]	[1,NE,0];[0,NE,1];[1,NE,0]
ESI3bmtR	[1,NE,0];[0,NE,1];[1,NE,0]	[0,NE,1];[1,NE,0];[0,NE,1]
ESI3btm	[0,NW,1];[1,NE,0];[0,NE,1]	[1,NE,0];[0,NE,1];[1,NW,0]
ESI3btmR	[1,NE,0];[0,NE,1];[1,NW,0]	[0,NW,1];[1,NE,0];[0,NE,1]
ESI3mbt	[0,NE,1];[1,NE,0];[0,NW,1]	[1,NW,0];[0,NE,1];[1,NE,0]

ESI3mbtR	[1, NW, 0]; [0, NE, 1]; [1, NE, 0]	[0, NE, 1]; [1, NE, 0]; [0, NW, 1]
ESI3mtb	[0, NW, 1]; [1, NW, 0]; [0, NE, 1]	[1, NE, 0]; [0, NW, 1]; [1, NW, 0]
ESI3mtbR	[1, NE, 0]; [0, NW, 1]; [1, NW, 0]	[0, NW, 1]; [1, NW, 0]; [0, NE, 1]
ESI3tbn	[0, NE, 1]; [1, NW, 0]; [0, NW, 1]	[1, NW, 0]; [0, NW, 1]; [1, NE, 0]
ESI3tbnR	[1, NW, 0]; [0, NW, 1]; [1, NE, 0]	[0, NE, 1]; [1, NW, 0]; [0, NW, 1]
ESI3tmb	[0, NW, 1]; [1, NW, 0]; [0, NW, 1]	[1, NW, 0]; [0, NW, 1]; [1, NW, 0]
ESI3tmbR	[1, NW, 0]; [0, NW, 1]; [1, NW, 0]	[0, NW, 1]; [1, NW, 0]; [0, NW, 1]
ESI4	[1, Cup, 0]; [0, NE, 1]	[0, Cup, 1]; [1, NW, 0]
ESI4R	[0, Cup, 1]; [1, NW, 0]	[1, Cup, 0]; [0, NE, 1]
ESI4t	[0, NW, 1]; [1, Cap, 0]	[1, NE, 0]; [0, Cap, 1]
ESI4Rt	[1, NE, 0]; [0, Cap, 1]	[0, NW, 1]; [1, Cap, 0]
ESI4tau	[1, Cup, 0]; [0, NW, 1]	[0, Cup, 1]; [1, NE, 0]
ESI4Rtau	[0, Cup, 1]; [1, NE, 0]	[1, Cup, 0]; [0, NW, 1]
ESI4tttau	[0, NE, 1]; [1, Cap, 0]	[1, NW, 0]; [0, Cap, 1]
ESI4Rtttau	[1, NW, 0]; [0, Cap, 1]	[0, NE, 1]; [1, Cap, 0]
ESI5	[1, Cup, 0]; [0, Cap, 1]	[1, Null, 0]
ESI5R	[1, Null, 0]	[1, Cup, 0]; [0, Cap, 1]
ESI5t	[0, Cup, 1]; [1, Cap, 0]	[1, Null, 0]
ESI5Rt	[1, Null, 0]	[0, Cup, 1]; [1, Cap, 0]
ESI6	[0, Cup, 0]; [0, Cap, 0]	[0, Null, 0]
ESI6R	[0, Null, 0]	[0, Cup, 0]; [0, Cap, 0]
ESI7	[0, Cap, 0]; [0, Cup, 0]	[2, Null, 0]
ESI7R	[2, Null, 0]	[0, Cap, 0]; [0, Cup, 0]
ESI8capdcupu	[0, Cap, 0]; [0, Cup, 0]	[2, Cup, 0]; [0, Cap, 2]
ESI8capdcupuR	[2, Cup, 0]; [0, Cap, 2]	[0, Cap, 0]; [0, Cup, 0]
ESI8capdneu	[0, Cap, 2]; [0, NE, 0]	[2, NE, 0]; [0, Cap, 2]
ESI8capdneuR	[2, NE, 0]; [0, Cap, 2]	[0, Cap, 2]; [0, NE, 0]
ESI8capdnwu	[0, Cap, 2]; [0, NW, 0]	[2, NW, 0]; [0, Cap, 2]
ESI8capdnwuR	[2, NW, 0]; [0, Cap, 2]	[0, Cap, 2]; [0, NW, 0]
ESI8capdu	[0, Cap, 2]; [0, Cap, 0]	[2, Cap, 0]; [0, Cap, 0]
ESI8capduR	[2, Cap, 0]; [0, Cap, 0]	[0, Cap, 2]; [0, Cap, 0]
ESI8cupdcapu	[0, Cup, 2]; [2, Cap, 0]	[0, Cap, 0]; [0, Cup, 0]
ESI8cupdcapuR	[0, Cap, 0]; [0, Cup, 0]	[0, Cup, 2]; [2, Cap, 0]
ESI8cupdneu	[0, Cup, 2]; [2, NE, 0]	[0, NE, 0]; [0, Cup, 2]
ESI8cupdneuR	[0, NE, 0]; [0, Cup, 2]	[0, Cup, 2]; [2, NE, 0]
ESI8cupdnwu	[0, Cup, 2]; [2, NW, 0]	[0, NW, 0]; [0, Cup, 2]
ESI8cupdnwuR	[0, NW, 0]; [0, Cup, 2]	[0, Cup, 2]; [2, NW, 0]
ESI8cupdu	[0, Cup, 0]; [2, Cup, 0]	[0, Cup, 0]; [0, Cup, 2]
ESI8cupduR	[0, Cup, 0]; [0, Cup, 2]	[0, Cup, 0]; [2, Cup, 0]
ESI8nedcapu	[0, NE, 2]; [2, Cap, 0]	[2, Cap, 0]; [0, NE, 0]
ESI8nedcapuR	[2, Cap, 0]; [0, NE, 0]	[0, NE, 2]; [2, Cap, 0]
ESI8nedcupu	[0, NE, 0]; [2, Cup, 0]	[2, Cup, 0]; [0, NE, 2]
ESI8nedcupuR	[2, Cup, 0]; [0, NE, 2]	[0, NE, 0]; [2, Cup, 0]
ESI8nedneu	[0, NE, 2]; [2, NE, 0]	[2, NE, 0]; [0, NE, 2]
ESI8nedneuR	[2, NE, 0]; [0, NE, 2]	[0, NE, 2]; [2, NE, 0]
ESI8nednwu	[0, NE, 2]; [2, NW, 0]	[2, NW, 0]; [0, NE, 2]

ESI8nednwuR	[2,NW,0] ; [0,NE,2]	[0,NE,2] ; [2,NW,0]
ESI8nwdcapu	[0,NW,2] ; [2,Cap,0]	[2,Cap,0] ; [0,NW,0]
ESI8nwdcapuR	[2,Cap,0] ; [0,NW,0]	[0,NW,2] ; [2,Cap,0]
ESI8nwdcupu	[0,NW,0] ; [2,Cup,0]	[2,Cup,0] ; [0,NW,2]
ESI8nwdcupuR	[2,Cup,0] ; [0,NW,2]	[0,NW,0] ; [2,Cup,0]
ESI8nwdneu	[0,NW,2] ; [2,NE,0]	[2,NE,0] ; [0,NW,2]
ESI8nwdneuR	[2,NE,0] ; [0,NW,2]	[0,NW,2] ; [2,NE,0]
ESI8nwdnwu	[0,NW,2] ; [2,NW,0]	[2,NW,0] ; [0,NW,2]
ESI8nwdnwuR	[2,NW,0] ; [0,NW,2]	[0,NW,2] ; [2,NW,0]

## References

- [1] J. Scott Carter and Masahico Saito, *Knotted Surfaces and Their Diagrams*. American Mathematical Society, Providence, 1998.