`coover`: glue between `heegaard` and `twister`

Robert C. Haraway, III

February 5, 2016

1 Introduction

The following problem has come to be important to a project I'm working on:

> Given a finite presentation P of a group G, to construct a triangulation of a 3-manifold M with fundamental group G.

There are various issues with this problem, not least of which is the undecidability of 3-manifoldness (indeed, even of triviality) from a finite presentation. For the purposes of the project, it will suffice to solve the following less impossible problem:

> Given a finite presentation P of a group G, to determine whether or not P is the associated presentation of a (generalized) Heegaard splitting of some 3-manifold, and if so, to construct a triangulation of the 3-manifold so split.

John Berge's program `heegaard`[1] can determine realizability of a presentation as a Heegaard splitting, and construct such a splitting if one exists. But it cannot construct triangulations.

Mark Bell, Tracy Hall, and Saul Schleimer wrote a program `twister` that can take as input a surface S and two attaching multicurves γ_0, γ_1, represented as a cubulation and two sets of disjoint hyperplanes, and triangulate the result of attaching 2-handles to $S \times [0, 1]$ along $\gamma_0 \times \{0\}$ and $\gamma_1 \times \{1\}$. If S has genus g and γ_0 has g components, then this is a generalized Heegaard splitting where the components of γ_0 are boundaries of compressing discs for the starting handlebody.

All that remains to solve the problem, then, is to turn the output of `heegaard` into a cubulation and list of hyperplanes.

I thank Mark Bell and Saul Schleimer for helpful conversations on how this could be accomplished.

2 Digestion and its results

The following is a list of presentations relevant to the aforementioned project.

$\langle$*heeg.in*$\rangle\equiv$

```
0
mGGmgNg
MNmn
```

To run `heegaard` on this input in a POSIX operating system, one places this file in the directory in which the `heegaard` executable is located; then one renames it `Input_Presentations`; then one runs `heegaard` from a command line; then one selects option ‘B’; then one selects ‘d’; then one replies ‘n‘ to further options. A session with `heegaard` following these directions went as below:

⟨*heegaard.out*⟩≡

```
$ ./heegaard


                                    HEEGAARD
                                  BY JOHN BERGE
                               jberge@charter.net
                                     2/5/15

 A PROGRAM FOR STUDYING 3-MANIFOLDS VIA PRESENTATIONS AND HEEGAARD DIAGRAMS.

        Copyright 1995-2015 by John Berge, released under GNU GPLv2+.

Note! Open the file 'Heegaard_Diagrams.dot' in Graphviz() to see Heegaard's Heegaard diagrams.

    Note!   Hit the 'space-bar' to interrupt long computations.
            Hitting 's' during long computations should provide a status report.

HIT 'B' TO DO SOME BATCH PROCESSING.
HIT 'f' IF THE PRESENTATION WILL COME FROM THE FILE 'Input_Presentations'.
HIT 'k' IF THE PRESENTATION WILL BE ENTERED FROM THE KEYBOARD.
HIT 'q' TO QUIT RUNNING THE PROGRAM.


Hit 'a' TO COMPUTE ALEXANDER POLYNOMIALS OF 2-GENERATOR 1-RELATOR PRESENTATIONS.
HIT 'b' TO FIND 'meridian' REPS M1 & M2 OF 2-GENERATOR 1-RELATOR PRESENTATIONS.
    (This allows one to check if such presentations are knot exteriors.)
HIT 'c' TO CHECK REALIZABILITY OF PRESENTATIONS.
HIT 'C' TO CHECK IF THE INITIAL PRESENTATION IS A "HS REP". (Heegaard will stop and alert the us
    a sequence of handle-slides of the initial presentation P yields a presentation P' with |P'|
HIT 'd' TO SEE DATA FOR HEEGAARD DIAGRAMS OF PRESENTATIONS.
HIT 'D' TO SEE THE DUAL RELATORS OF EACH REALIZABLE PRESENTATION'S DIAGRAM.
HIT 'E' TO HAVE HEEGAARD STABILIZE THE IP, COMPUTE HS REPS AND CHECK IF THE IP APPEARS ON THE HS
HIT 'h' TO FIND THE INTEGRAL FIRST HOMOLOGY OF PRESENTATIONS.
HIT 'l' TO FIND THE SIZE OF ORBITS OF PRESENTATIONS UNDER LEVEL TRANSFORMATIONS.
HIT 'q' TO QUIT RUNNING IN BATCH MODE.
HIT 'r' TO REDUCE AND SIMPLIFY PRESENTATIONS USING DEPTH-FIRST SEARCH AND SEP_VERT SLIDES.
HIT 'R' TO REDUCE AND SIMPLIFY PRESENTATIONS USING BREADTH-FIRST SEARCH AND SEP_VERT SLIDES.
HIT 's' TO FIND SYMMETRIES OF PRESENTATIONS.
HIT 'S' TO CONVERT SNAPPY FORMAT PRESENTATIONS TO HEEGAARD READABLE PRESENTATIONS.
HIT 'u' TO STABILIZE PRESENTATIONS WHILE PRESERVING REALIZABILITY.
HIT 'x' TO SIMPLIFY PRESENTATIONS BY SUCCESSIVELY DELETING PRIMITIVES, WITHOUT CHECKING REALIZAB
HIT 'X' TO FIND PRESENTATIONS OBTAINED BY DELETING PRIMITIVES FROM ONLY INITIAL PRESENTATIONS.
HIT 'z' TO REDUCE PRESENTATIONS TO MINIMAL LENGTH.


SHOW WHICH FACES OF EACH DIAGRAM FORM BDRY COMPONENTS ?  HIT 'y' OR 'n'.

PRINT DUAL RELATORS OF EACH DIAGRAM ? HIT 'y' OR 'n'.
```

```
PRINT PATHS CONNECTING FACES OF EACH DIAGRAM ? HIT 'y' OR 'n'.

------------------------------------

0
    mGGmgNg
    MNmn

L 11, Gen 3, Rel 2.

Rewrote the presentation using the substitution: CbA.

 The rewritten presentation is:
    AABaCaB
    BCbc


I) The following table gives the number of edges joining each pair of vertices.

(A --> a1,B1,b1,C1), (a --> B1,b1,c1), (B --> C1,c1), (b --> C1,c1)

For each (X,x) pair of vertices with (X,x) = (A,a), (B,b) ... ,(Z,z):
1) Number the edges at vertex X counter-clockwise about vertex X giving the 'first-edge' at vert
2) Note the 'first-edge' at vertex V is the first edge in counter-clockwise order about V which
3) For x = a,b ... ,z, number the edges at vertex x clockwise about x, giving the 'first-edge' a

(1,0,0)

II) Vertices in the boundary of each face of the Heegaard diagram in clockwise order are:

F1) AaB, F2) ABC, F3) ACb, F4) Aba, F5) abc, F6) acB, F7) BcbC

Note: Heegaard chose the cycle 'BcbC' to be the boundary of the 'infinite' face.

III) CO[v] lists the vertices in the link of vertex v in counter-clockwise cyclic order starting

CO[A] = abCB, CO[a] = ABcb, CO[B] = ACca, CO[b] = AacC, CO[C] = AbB, CO[c] = aBb

****************************************************************************************
The following lines describe the Heegaard diagram in Graphviz() readable form.
Copy and paste into 'Heegaard_Diagrams.dot to have Graphviz() display the diagram.

 graph G{layout = neato; model = circuit; size = "10.0,8.0"; ratio = fill ;
 label = "Diagram of Presentation 1 of the Initial Presentation 0";
node [shape = circle, fontsize = 10, height = 0.1, style = white]
A [pos = "240,262!"]; a [pos = "339,187!"]; B [pos = "30,30!"]; b [pos = "550,420!"]; C [pos = "
 edge [fontsize = 10]; { A -- a ; A -- B ; A -- b ; A -- C ; a -- B ; a -- b ; a -- c ; B -- C ;
```

```
 Totals: NumPresExamined 1

HIT 'B' TO CONTINUE IN 'BATCH' MODE. HIT 'q' TO QUIT RUNNING IN BATCH MODE.

HIT 'B' TO DO SOME BATCH PROCESSING.
HIT 'f' IF THE PRESENTATION WILL COME FROM THE FILE 'Input_Presentations'.
HIT 'k' IF THE PRESENTATION WILL BE ENTERED FROM THE KEYBOARD.
HIT 'q' TO QUIT RUNNING THE PROGRAM.
```

It turns out that the only information we need from `heegaard` is, for each presentation P,

- the presentation that `heegaard` transforms P into;
- the table from part I;
- the tuple from part I; and
- the string array `CO` from part III.

Culling the extra output is an exercise in regular expressions. The following is a sequence of `sed` regular expressions that accomplishes this for the above output:

$\langle$*cull.sed*$\rangle\equiv$

After this, we need to rewrite this all in a format that can be interpreted in Python (for `twister` is a Python module). The following `sed` script does this.

$\langle$*rewrite.sed*$\rangle\equiv$

The result is the following file of data ready for input to Python.

$\langle$*digested.py*$\rangle\equiv$

3 An example worked by hand

4 Extraction: the core

4.1 Neighbor logging suffices...

4.2 The inner loop: or, the next flag

4.3 ...but we can do better: naming cycles

5 Extraction: the rest

References

[1] Berge, John. `heegaard`. Available at `http://www.math.uic.edu/t3m/`.

[2] van Gasteren, A. J. M. *On the shape of mathematical arguments.* With a foreword by Edsger W. Dijkstra. *Lecture Notes in Computer Science*, **445**. Springer-Verlag, Berlin, 1990.