Dear Potential ***CirclePack*** user:

CirclePack is an open source Java program for creating, manipulating, analyzing, storing, and visualizing circle packings. The author is Ken Stephenson, now Professor Emeritus at the University of Tennessee, Knoxville. The latest version "CirclePack-Jxx.jar" is available through a link at circlepack.com; changes are still occurring in the code, so it's best to check there for the latest version. (See also github for the source.)

A circle packing is a configuration of circles with a specified pattern of tangencies. The best source for background is "*Introduction to circle packing: the theory of discrete analytic functions*", by Ken Stephenson, Camb. Univ. Press, 2005, ISBN 978-0-521-82356-2. In addition to providing a basis for discrete analytic function theory and several other topics in pure mathematics, it has found concrete applications in areas from 3D printing, brain imaging, and physics to biology, as well as in art and architecture.

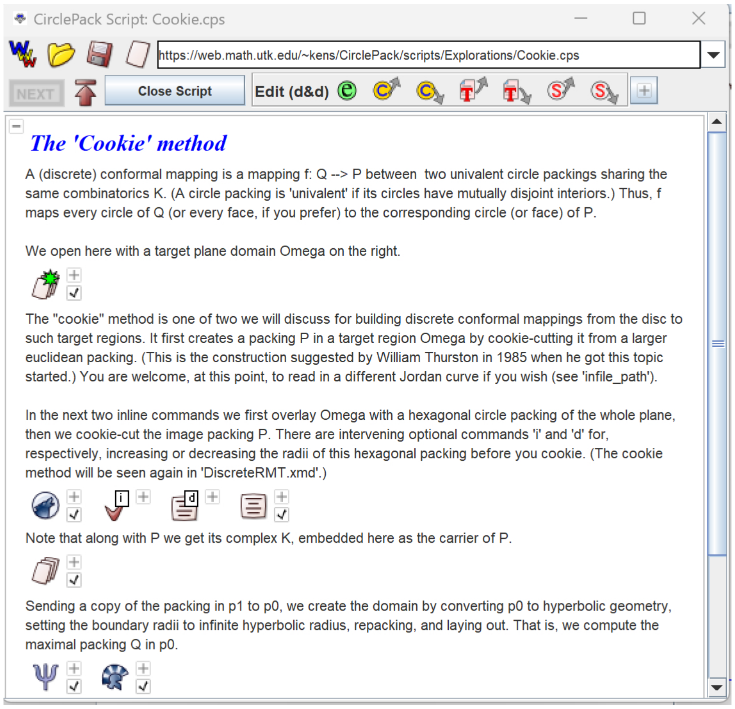
CirclePack is an experimental laboratory and seems to be the most comprehensive package available for circle packing. It does, however, take some effort to learn. Here are a few main points to consider if you want to try it out:

- CirclePack-Jxx.jar should open and run on almost any platform, though you may have to update Java. Try "java -jar CirclePack-Jxx.jar".
- ***CirclePack*** uses individual commands for carrying out all its actions. It is also graphics based, so there are several windows, menus, help files, canvasses, popup windows, etc., where actions can often be carried out via the mouse. However, there are also several "command:" locations where commands can be entered manually.
- The running program accommodates three circle packings at any one time, each associated with its own display canvass; one of these is the "active" packing, and thus the default target of commands.
- Circle packing tangency patterns are encoded in graphs, in particular, triangulations of open or closed topological surfaces. The underlying geometry may be euclidean, hyperbolic (unit disc model), or spherical.
- The program's control panel has a "Help" button which opens a window containing several tabs. The "Command Details" tab describes all the individual commands in alphabetical order. E.g., the command "*disp -w –cf a –f 7*" will clear the active canvas, draw all its circles as filled circles, and draw face number 7.
- There are various user files associated with ***CirclePack:***
  - Data files: These are typically "*.p" files and store formatted descriptions of circle packings, the triangulation graph of a packing, its circles' radii, centers, and perhaps various other data --- typically, everything necessary to restore a packing in a future session.

- Scripts: These are typically “*.cps” (or “*.xmd) files and contain prepared command strings and possibly data files. A properly prepared script will allow any user to run through a full ***CirclePack*** session by simply clicking through the script.
- Image files: ***CirclePack*** can export circle packing images in various formats, including *.jpg and *.ps.

I consider scripts as essential for learning and using ***CirclePack.*** A command string is a semi-colon separated sequence of commands, exactly like commands one could type into a command line from the keyboard. A script file is mainly a sequence of such prepared command strings, each of which can be executed with a single click, but also allows for text sections to explain the action and data files to be loaded.

Scripts are loaded/saved/edited via the “CirclePack Script” window. An image of a loaded script window is provided below. Existing scripts can be called in from the local computer or from the web, and there’s an icon that opens an empty script for editing. When running ***CirclePack***, you can familiarize yourself with script window features by hovering the mouse over the various icons to bring up hints.


CirclePack Script: Cookie.cps
https://web.math.utk.edu/~kens/CirclePack/scripts/Explorations/Cookie.cps
NEXT
Close Script
Edit (d&d)

The 'Cookie' method

A (discrete) conformal mapping is a mapping f: Q --> P between two univalent circle packings sharing the same combinatorics K. (A circle packing is 'univalent' if its circles have mutually disjoint interiors.) Thus, f maps every circle of Q (or every face, if you prefer) to the corresponding circle (or face) of P.

We open here with a target plane domain Omega on the right.

The "cookie" method is one of two we will discuss for building discrete conformal mappings from the disc to such target regions. It first creates a packing P in a target region Omega by cookie-cutting it from a larger euclidean packing. (This is the construction suggested by William Thurston in 1985 when he got this topic started.) You are welcome, at this point, to read in a different Jordan curve if you wish (see 'infile_path').

In the next two inline commands we first overlay Omega with a hexagonal circle packing of the whole plane, then we cookie-cut the image packing P. There are intervening optional commands 'i' and 'd' for, respectively, increasing or decreasing the radii of this hexagonal packing before you cookie. (The cookie method will be seen again in 'DiscreteRMT.xmd'.)

Note that along with P we get its complex K, embedded here as the carrier of P.

Sending a copy of the packing in p1 to p0, we create the domain by converting p0 to hyperbolic geometry, setting the boundary radii to infinite hyperbolic radius, repacking, and laying out. That is, we compute the maximal packing Q in p0.

One way to learn the command structure is to run through existing scripts. Several are available here; load them using the internal web browser which comes up using the “WWW” icon. Once you have loaded a script, do a quick run through by clicking the “Next” button and watching the action. Then do another, slower run through: open a command item using the “+” button and see its actual string of commands. You can type them one at a time into a “command:” line if you want to see the effect of each individual command. Visit the “Help” file to view the details behind each command.

Once you want to begin your own work, it is recommended that you proceed using the script window: Open a new command item, enter your command, and click the “try” icon to execute it. If the command doesn’t work, adjust it and “try” again. Add semi-colon separated commands as you progress. Save the script so you can return and easily get back to where you left off. You can also intersperse text blocks for comments, and there’s a data section so that files can be included. If you want to share a ***CirclePack*** session with a collaborator, you can simply organize it in a script, save it, and send it as a single file.

Your particular interest in circle packing may involve existing data (typically combinatorial) from some outside source. To read that data, you should check out the “*read <filename>*” command in the “Help” window. Also look at the “Format” tab to see the various data formats that ***CirclePack*** can read.

If you have questions, I’m open to corresponding since I enjoy seeing ***CirclePack*** being used.

Ken Stephenson, June 2024