Introduction to the PascGalois Applet System

Currently there is one main application, PascGalois JE, and 17 applets. Over the past three years we have added more and more features to the PascGalois JE program and even though we have redesigned the user interface twice the program has become a bit overwhelming for beginning students. So we decided that a good pedagogical step would be to develop a sequence of applets that exposed the students to the features a little at a time. We also decided to rewrite the sequence of Abstract Algebra laboratories into an entirely web-based format with the needed applets built directly into the lab. This way your students would not need to install any software, nor would your IT department need to install anything in your computer labs. As long as they have the current version of the Java JRE installed and a web browser your students will be able to do the labs.

The applets use the same syntax as the main program and have a very similar look and feel to the user interface. We have found that our Abstract Algebra students prefer to use the applets to do the labs but for undergraduate research we would suggest that the student use the PascGalois JE program so that they can utilize some of the more powerful features that are not available in the applets.

If you find that you would rather use the PascGalois JE program in your classes instead of the applets we have made the installation of the software as painless as possible. If you use the Windows operating system we provide an install program that will install the PascGalois JE program along with all of the third party software that is needed. You will still need to install the current version of the Java JRE. Also if you are on a Windows system we provide an ISO image that will allow you to burn a stand-alone PascGalois CD, which will run the program on any Windows PC regardless of its version of the Java JRE. If you are using a Linux/Unix or Mac systems we have made the installation as painless as possible.

The PascGalois JE Program

PascGalois JE is a platform independent multiple document interface Java application for exploring one and two dimensional cellular automata over finite group structures. It currently supports the integers under addition mod n, the integers under multiplication mod n, the symmetry group for a regular n-gon, the Quaternions, the generalized Quaternion groups, dicyclic groups, and the group of permutations on n letters. Furthermore, there is an advanced mode that allows the user to work with arbitrary products and quotients of these structures. The program also has a facility where the user can input their own structure via an operation table.

The program allows the user to alter color schemes, zoom in and out on portions of the image, select regions for element counts, period and death calculations of finite automata, three dimensional viewing as well as level and density graphing modes for two dimensional automata, animation options and POV-Ray export facilities and, of course, file saving and loading of program information.

The PascGalois Applets

The sequence of applets is divided into four sections. The Single Group Viewers, created by Katie Ford, include only the most basic options for viewing one-dimensional cellular automata. These applets focus on exploring a single class of group structure. The Viewers with Group and Seed Options, created by Israa Taha, allow the user to select the group structure and enter more complicated seeds. It also allows the user to use the advanced group structure mode and has a facility for user-defined structures. The Viewers with Group, Seed and Update Rule Options, created by John Zimmerman, adds several more options along with the ability to alter the update rule. Finally, the Viewers with Full Options add the ability to do element counting and include the group calculator. There are two other applets as well. One is simply the group calculator for doing group operations and generating subgroups and cosets and the other is a superimposer applet that has a specialized function that is used in one of the Abstract Algebra labs.

All of the applets have associated applications that can be downloaded and installed on the user's local machine. These files can be downloaded from each individual page or the application download page. Also, each applet has the option of running full screen which will open the applet up in its own window and. To run the applet in full screen mode simply click on the Full Screen link below the applet.

Single Group Viewers

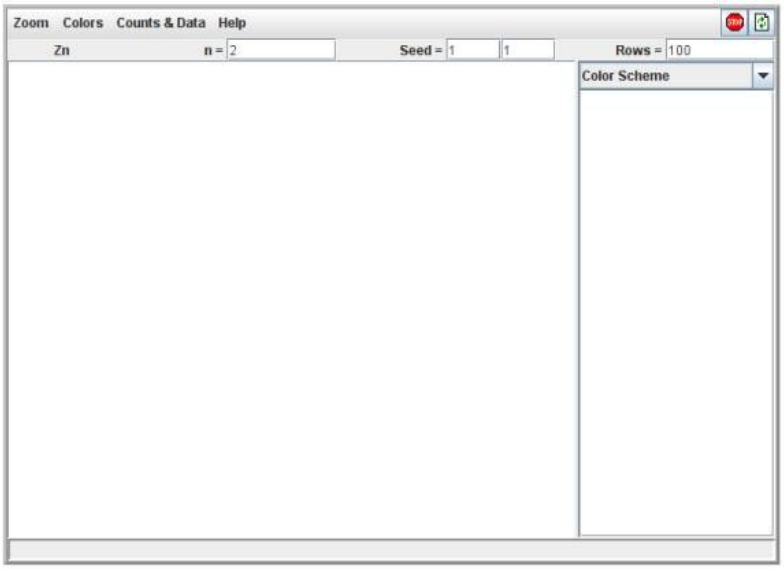
There are 8 single group viewer applets, one for each of the group classes that the PascGalois JE program supports. These applets are Z_n, U_n, Z_n X Z_m, D_n, S_n,, Q, Q_n, and C_n. All of these applets offer the same set of functions. We will do a few exercises with the Z_n applet and then briefly discuss the element syntax for the other group structures.

The first applets are restricted to graphing one-dimensional automata using the Pascal's triangle update rule over only one class of groups. The applets in this series have facilities for color scheme alteration, zooming and element counting.

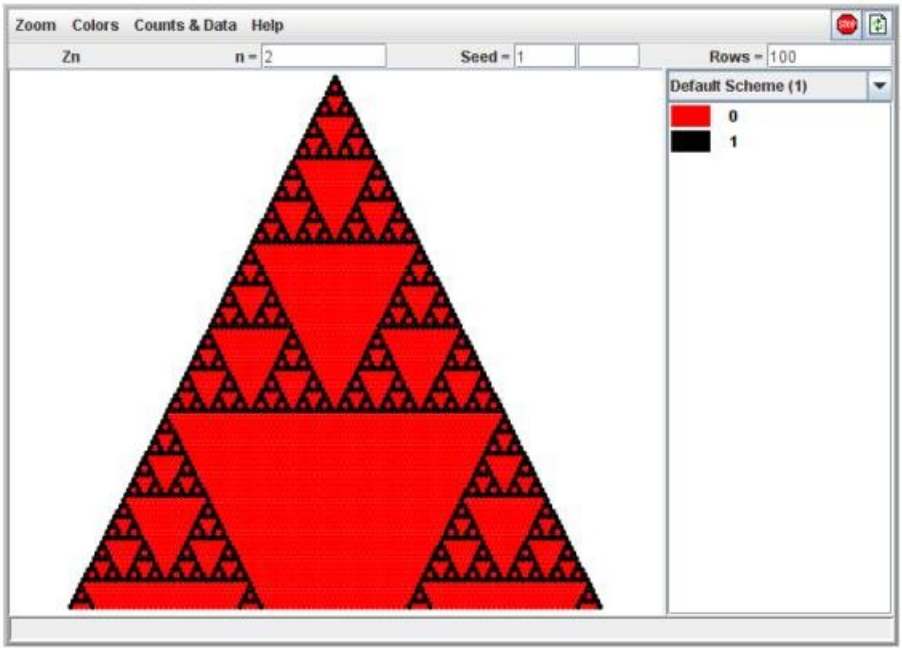
Some basics about triangle generation

We are going to start out with Pascal's triangle modulo 2, 3, 4, 5 and 6.

1. Go to the web pages on the CD and click on the PascGalois JE - Applets link.
2. On the menu on the left under Single Group Viewers select the Z_n menu item. At this point you should see the Z_n applet, pictured below.



3. We will start with Z_2. Since this applet graphs Z_n we need to input the value of 2 for n. At the top of the applet you will see an "n =" box simply input a 2 into this box. (2 is the default entry so you will probably not need to do anything here)
4. We also need to give the program seed values. This series of applets uses a default of two seeds, mainly because most of the groups encountered in an introductory abstract algebra course have either one or two generators. We can force the program to use only one seed by leaving one of the seed entries blank. So to start with a seed of just a 1 (like in Pascal's triangle) we would put a 1 in one of the seed boxes and leave the other box blank. Change the seed entries so that there is a 1 in one box and nothing in the other.
5. The number of rows tells the program how many rows (or time-steps) of the automaton should be generated *after* the seed row. So if this is set to 100 the image will actually contain 101 rows. We will leave this entry at 100.
6. The creation and graphing of one of these triangles can be a little time consuming if there are a large number of rows to create, so when we wrote the software we did not have the automaton regraph every time a change was made. So to create or update an image you must click the Refresh/Apply tool button in the upper right corner of the applet. Try this. Notice that the image appears in the box on the left and the color correspondence in the box on the right. Also note that the division bar between these two boxes is movable.



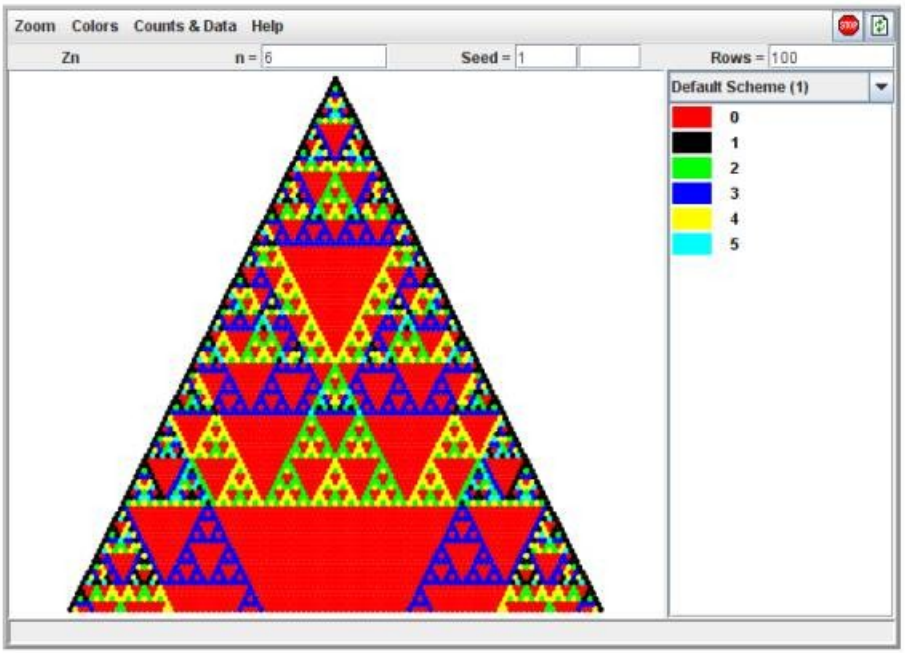
7. Move your mouse over the image and notice what appears in the status bar at the bottom. The program tells you what location the element is in as well as what the element is. Furthermore, if you hover over a location for a second or two a tool tip will appear with the same information.
8. Change *n* to 3, 4, 5 and 6 and regraph the image in each case to see the different patterns. These images serve as "signatures" for the cyclic group structures. In the lab sequence the students will encounter these images in different contexts and it is hoped that they will recognize these signatures so that they can make statements about isomorphic structures.

Some basics about colorings

Often our choice of coloring affects the type and/or amount of structure we observe in Pascal's and other related triangles. We will see that altering the colors often reveals hidden structure in the images.

When you generate a triangle the program will use the default color settings and color each element a unique color (up to 60 elements and then it rotates the color scheme). The program allows you to change the color of any element as well as group sets of elements together with the same color. We will look at a few examples below. There is also a feature where you can drag and drop colors from one window to another. Unfortunately, with the applets you will not be able to save your color schemes since applets do not have access to the user's hard drive. The PascGalois JE application does have the ability to save and load the color schemes that you create.

1. Generate Pascal's triangle modulo six, keep the number of rows at the default 100.



2. Just to see how to change the color of an individual color, do the following. Double-click the element 3 either on the image or the element list. At this point the color chooser dialog box will appear. Select a purple color and click OK. You will notice that the color has changed in the color correspondence box. Now click the Refresh/Apply toolbar button. Notice what happened to the triangle image.
3. To reset the colors to the default scheme select the Colors > Reset to Default Color Scheme in the menu.
4. Now we will highlight colors. Select both 3 and 5 from the color correspondence. To select multiple items simply hold down the Control key and click all the items of interest. Now select Colors > Highlight Elements from the menu. You will notice that the elements that were selected are now colored red and the other elements are black. Let's change these colors before refreshing the image. Double-click either the 3 or the 5 and select the color yellow. Now double-click any of the black colors and select a gray color. Notice that all of the colors in the respective groups changed when you made a single change. This is because the elements are now linked together. The 3 and 5 are considered a set and the 0, 1, 2 and 4 are a set. Refresh the image.
5. To ungroup the colors select Colors > Reset to Default Color Scheme from the menu and refresh the image.
6. We will use another type of color grouping, subset grouping. In the color correspondence window select the numbers 0 and 3. Select Colors > Group Elements from the menu. Note that 0 and 3 are now the same color. Now select 1 and 4 and then select the Colors toolbar button followed by Group Elements. Finally select 2 and 5 followed by the Colors toolbar button and then Group Elements. Refresh the graph.

7. You probably noticed that what we really did in the last exercise was color all of the cosets of $Z_6/\langle 3\rangle$, in other words we got a signature for the group $Z_6/\langle 3\rangle$ whis surprisingly enough looked a lot like the signature for Z_3. I smell an isomorphism here. Exercises like this one are in the quotient group labs. Although the process we used in the above exercise is good pedagogically, having them physically identify every member of the coset with the same color (in effect making the coset a single element) it can be tedious with larger examples. So we wrote in a special coloring feature that will automatically color cosets. Reset the colors to the default scheme and refresh the image. Now select the element 3 in the color correspondence list on the right and select Colors > Group Left Cosets. At this point you should get the same color scheme as you did in the last example. What this option really does is it takes all of the selected elements in the color scheme window, generates the subgroup generated by these elements, calculates the cosets of this subgroup and then colors the elements of each coset the same color using the default coloring scheme.
8. Reset the colors to the default scheme and refresh the image.
9. You can also use the PascGalois triangle itself to select colors. Put the mouse over a section of red (the element 0) and double click. The color selector will appear for the element 0. Select a different color and click OK. Note that the new color is in the color correspondence box. Refresh the image to see the new triangle.
10. Another way to change an element's color is to right-click on the element either in the color correspondence box or on the image and a small popup menu will appear with two options, Set Element Color and Set Element Color to Transparent. If you select the transparent option the color box will simply be a rectangle with an X through it. Refresh the image to see the change. You can also change the background color by selecting Colors > Set Background Color.
11. A few other things to note about color changes. There are options to undo and redo color scheme changes. The program will keep up to 20 changes for each color scheme. There are also facilities to add, remove and rename color schemes. If you do add color schemes you can select the different color schemes using the drop-down selector over the color scheme window.

Some basics about zooming

If you don't have Pascal's triangle modulo six on the screen please regenerate it, keep the number of rows at the default 100.

1. Select Zoom > Zoom In from the menu. Notice that the mouse pointer has changed when you are in the triangle window. Click somewhere inside the triangle. Click several more times to see what happens.
2. Select Zoom > Zoom Out from the menu. Notice that the mouse pointer has changed again. Click somewhere inside the triangle. Click several more times to see what happens.
3. Select Zoom > Reset Zoom to Full View from the menu. Notice that the mouse pointer has not changed but the triangle has zoomed out to its fullest.
4. Select Zoom > Turn Off Zoom from the menu. Notice that the mouse pointer has changed back

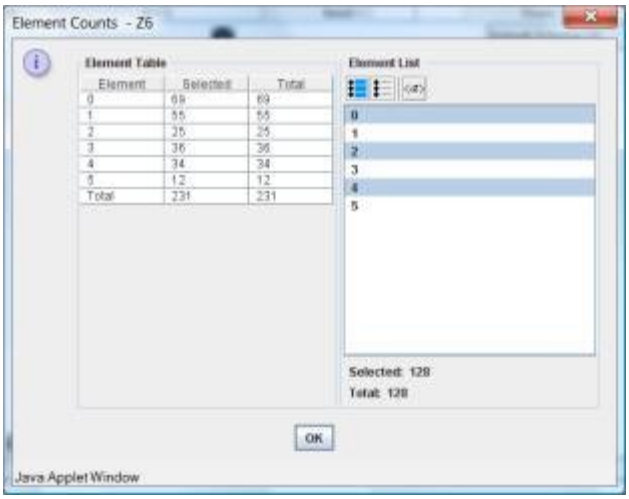
to its default.

5. The default zoom is 2X. This can be changed by selecting Zoom > Zoom Factor from the menu followed by the desired zoom factor.
6. Select Zoom > Zoom Box from the menu.
7. Click and drag over a portion of the triangle. Note that the area will be shaded. Release the mouse button and the program will zoom in on the selected portion. The program may need to alter the bounds of the selection but you will get at least what you selected.
8. If you are in the process of zooming with the Zoom Box feature and notice that your area is not what you want you can cancel the zoom by pressing the right mouse button before releasing the left button. Give this a try.
9. Reset the zoom to full view and turn the zooming off.
10. There are also facilities to undo and redo zooms. The program will keep a maximum of 50 zooms in memory.

Some basics about element counting

These applets also have a feature for counting the number of elements within a given selected region. This option was put in mainly for fractal dimension explorations in a dynamical systems course. Although sequence of Abstract Algebra labs do not use this feature, we will take a quick look at this facility.

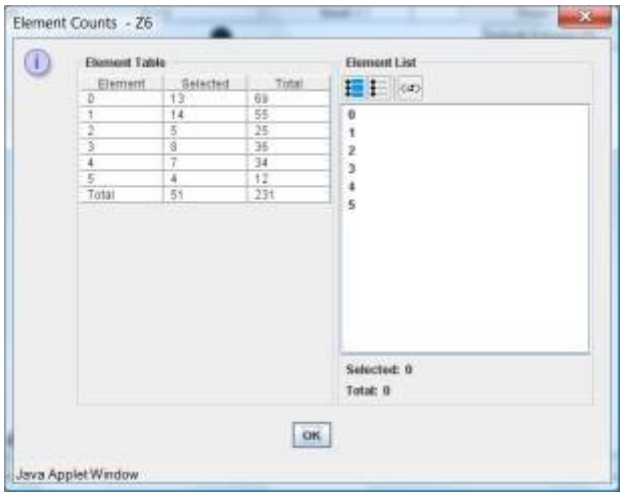
1. Graph Z_6 again with the default color scheme and use 20 rows (really 21).
2. Select Counts & Data > Select All. Notice that all of the elements are now highlighted. Select Counts & Data > Display Counts….



3. The left side gives the selected and total counts for each element. The selected counts are for the elements that are highlighted and the total counts are for all of elements that were graphed, since we selected all of the elements these counts are the same. On the right there is a list of

group elements. If you select one or more of these the program will add up the counts for each element and display the results below. There are toolbar buttons to select all, select none and to select a subgroup.

4. Click the OK button to close the counts window. Go back to the applet and select Counts & Data > Select None to clear the highlighting.
5. Select Counts & Data > Select Rows… A dialog box will appear allowing you to input a beginning row and ending row. Input 5 and 10 and click the OK button. You should see rows 5 to 10 highlighted. If you select the Display Counts option at this time you should get the following.



6. You can unselect rows as easily with the Counts & Data > Unselect Rows… option. Do so and unselect rows 6 to 7. At this point rows 5, 8, 9, and 10 should be selected.
7. Select Counts & Data > Select None to clear the selections.
8. The applets also allow you to select triangular regions using the mouse. Select Counts & Data > Select Region, note the change in the cursor. It looks like a selected triangle with a small rectangular box at the bottom. This rectangular box is actually a progress bar. Click on position (8, 0), note that the cursor has changed slightly, there is some red in the little progress bar indicating that you have selected one position. Now move to position (8, 8), note that as you do there is a line connecting the cursor with the point you selected. Click on position (8, 8). Note the progress bar is a little further indicating that you have selected two points. Now move to position (16, 8), note that as you do there is a shaded triangle connecting the cursor with the points you selected. Click on position (16, 8). At this point the triangle should be highlighted. The Unselect Region option works in the same manner.
9. The above selection could also be made using the Select Region by Positions option. If you select this from the Counts & Data menu a dialog box will appear allowing you to input the three vertices of the triangular region you want. If you input the three vertices we used above you will get the same region.
10. There are also facilities for changing colors and undoing and redoing selections.

Other applets in this first series

We also have applets in this first series to generate triangles over U_n (the integers under multiplication mod n), Z_n X Z_m (the Cartesian product of Z_n and Z_m both under addition) D_n (symmetry group for a regular n-gon), Q (the Quaternions), S_n (group of permutations on n letters), Q_n (generalized Quaternion groups) and C_n (the dicyclic group). These applets have the same features as the Z_n applet, the only difference is in the notation for the seed values. We give a short description of the supported groups and their syntax below.

Z_n: Integers under addition mod n.

Z_n is the set of elements {0, 1, 2, ... , n-1} under the operation of addition mod n. For example, in Z_5, 4+3 = 2. The program syntax for elements in this group are simply non-negative integers. Any input greater than or equal to n is replaced with itself mod n. So if you are working with Z_5 and use some input of 12 it will be converted to 2.

Z_n X Z_m: Cartesian product of the Integers under addition mod n and mod m.

Z_n X Z_m is the set of all ordered pairs of elements from {0, 1, 2, ... , n-1} and {0, 1, 2, ... , m-1} under the operation of componentwise addition mod n and m respectively. For example, in Z_5 X Z_7 (4, 2) + (3, 1) = (2, 3). The program syntax for elements in this group are just like we would write them mathematically. You must start with a (then an element of Z_n followed by a comma, then an element of Z_m and finally ending with a). Any component greater than or equal to the respective modulus is replaced with itself mod n, or m.

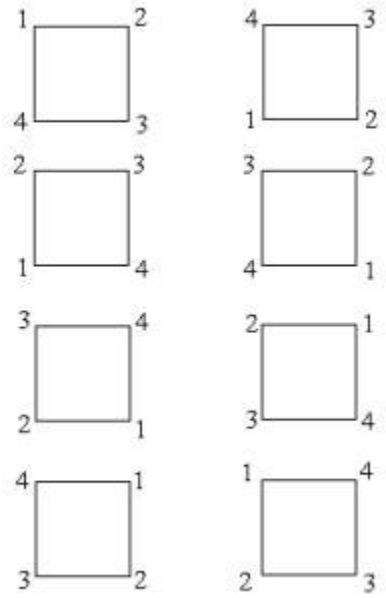
U_n: Integers under multiplication mod n.

U_n is the set of elements {0, 1, 2, ... , n-1} under the operation of multiplication mod n. Note that the elements used in the program need not be relatively prime to n. For example, in U_6, 2*3 = 0. The program syntax for elements in this group are simply non-negative integers. Any input greater than or equal to n is replaced with itself mod n. So if you are working with U_5 and use some input of 12 it will be converted to 2.

D_n: Symmetry group for a regular n-gon.

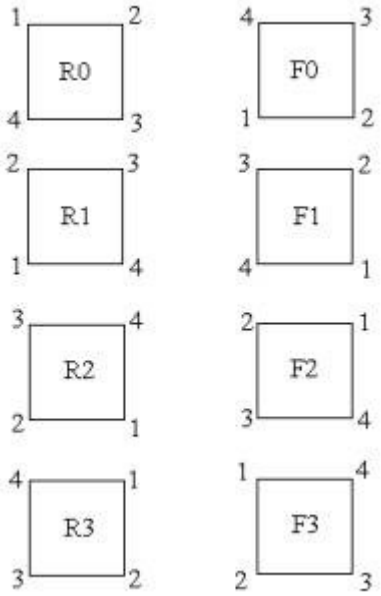
D_n is the set of all symmetry transformations of a regular n-gon, also known as the Dihedral group. Hence D_n contains 2n elements, n rotations and n reflections or flips. The operation is composition of

transformations. For example, D_4 is the symmetry group of the square. There are four rotations (0, 90, 180, and 270 degrees) and four flips (one over the horizontal, one over y=x, one over the vertical and one over y=-x). Graphically we can represent the elements of this group by the following figure. The original square is in the upper left. The rotations are down the first column and the flips are down the second column.



When representing these elements in the computer we can not easily use degree or radian measurement for rotations, especially when the angle does not divide evenly into 360 (for example, with D_7). So we have adopted the following scheme. R0 represents rotation by 0 degrees, that is, the identity element of D_n. R1 represents the first rotation that is a symmetry. So in D_4, R1 represents rotation by 90 degrees, in D_5 R1 represents rotation by 72 degrees and in D_6 R1 represents rotation by 60 degrees. In general, in D_n R1 represents rotation by 360/n degrees. R2 is the second such rotation, that is, rotation by 360*2/n degrees and so on up to R(n-1). The flips are represented by F0, F1, ..., F(n-1). F0 is the flip over the horizontal. F1 is the first possible flip that is a symmetry. In in D_4, F1 represents the flip over the line that is 45 degrees from the horizontal, in D_5 F1 represents the flip over the line that is 36 degrees from the horizontal, and in D_6 F1 the flip over the line that is 30 degrees from the horizontal. And so on. All of our angle measurements are counterclockwise, of course.

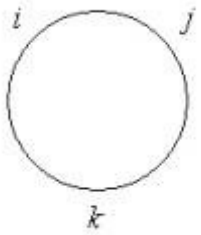
So from our above example of D_4, we have,



The operation for this group is composition of functions. We use the right to left convention of composing functions. For example, R1 * F2 = F3 and F2 * R1 = F1.

Q: The Quaternions.

The Quaternions is a noncommutative group of order 8 consisting of the elements {1, -1, i, -i, j, -j, k, -k} with $i^2 = j^2 = k^2 = -1$ and a circular definition for multiplying i, j, and k. If we consider the following circle with i, j, and k on it. To find the product of i * j we start at i and go around the circle to j (using the smallest arc) then we keep going to the next letter, which is of course k. Now if we moved in a clockwise direction we use a positive k and if we had to go counterclockwise we use a -k.



So we see that, i*j = k, j*i = -k, j*k = i, k*j = -i, ... Multiplication by 1 and -1 is the same as with integers, -1 * i = -i and so on.

The Quaternions can also be defined as the group of order 8 having two generators x and y satisfying the relations $x^2 = y^2$ and xyx = y. Since the first representation is more common in introductory abstract algebra classes we use it in this program. The element syntax for the program is simply 1, -1, i, -i, j, -j, k, -k, where i, j, and k can be lower or upper case.

S_n: Group of permutations on n letters.

S_n is the group of permutations on n letters, also called the symmetric group on n letters. There are several ways to represent the elements of this group but we have adopted the cycle notation since it is easier to input from a keyboard and is a little faster to do the necessary calculations. For example, if we are working in S_7 the permutation (2 3 5 4 7) means that 2 is sent to 3, 3 is sent to 5, 5 is sent to 4, 4 is sent to 7, 7 is sent to 2, and both 1 and 6 are fixed. The identity element is represented by (1). The operation for this group is composition and as with the Dihedral groups we do composition from right to left. For example,

(2 3 5 4 7)(1 2 3 5 6) = (1 3 4 7 2 5 6) and (1 2 3 5 6)(2 3 5 4 7) = (1 2 5 4 7 3 6)

The program syntax for S_n is just how it is written mathematically. A cycle must start with a (have numbers between 1 and n separated by at least one space and ending with a). For permutations that are composed of two or more disjoint cycles we simply use juxtaposition. For example, (1 3 2)(4 6 5). The program outputs cycles in a standard form with the smallest numerical entry in the cycle at the beginning. As a user, you do not need to input cycles in that form. For example, (1 3 2) can be input as (3 2 1) or as (2 1 3).

Q_n: Generalized Quaternion groups.

Q_n is the generalized quaternion group, n > 2. It is a group of order 2^n generated by two elements a and b under the following relations.

$$a^{2n-1} = 1,\ bab^{-1} = a^{-1},\ \text{and}\ b^2 = a^{2n-2}.$$

The element syntax for this group is any string of a and b to positive powers. For example, the user can input aba, babbab, a^2b^3, b^2a^3, and so on. Since every element of the generalized quaternions can be rewritten in the form a^tb^r with 0 <= t <= 2^{n-1} and 0 <= r <= 1 the program will simplify every element to this form.

C_n: Dicyclic group.

C_n is the dicyclic group, n > 1. It is a group of order 4n generated by two elements a and b under the following relations.

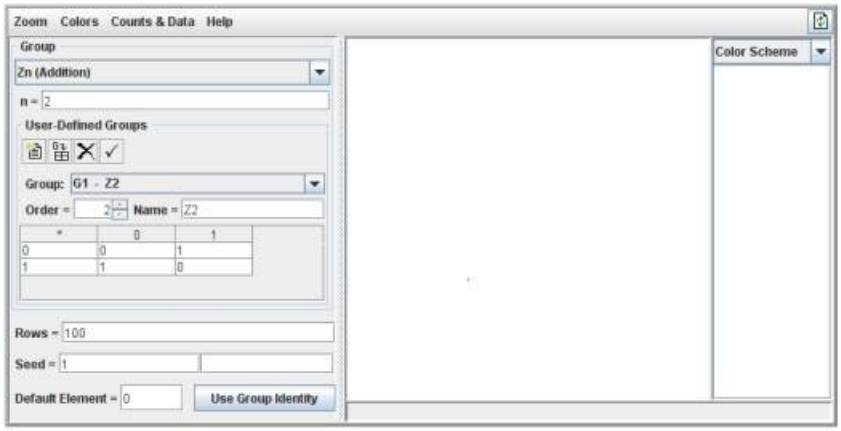
$$a^{2n} = 1,\ b^{-1}ab = a^{-1},\ \text{and}\ b^2 = a^n.$$

The element syntax for this group is any string of a and b to positive powers. For example, the user can input aba, babbab, a^2b^3, b^2a^3, and so on. Since every element of the dicyclic group can be rewritten in the form a^tb^r with 0 <= t <= 2n-1 and 0 <= r <= 1 the program will simplify every

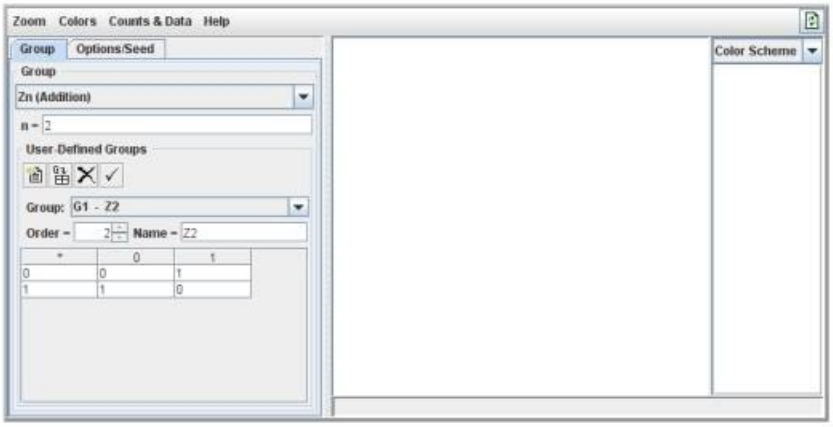
element to this form. One thing to note is that when n is a power of 2 the dicyclic group is isomorphic to a generalized quaternion group. More specifically, C_2^n is isomorphic to Q_{n+2} and C_2 is isomorphic to the quaternion group.

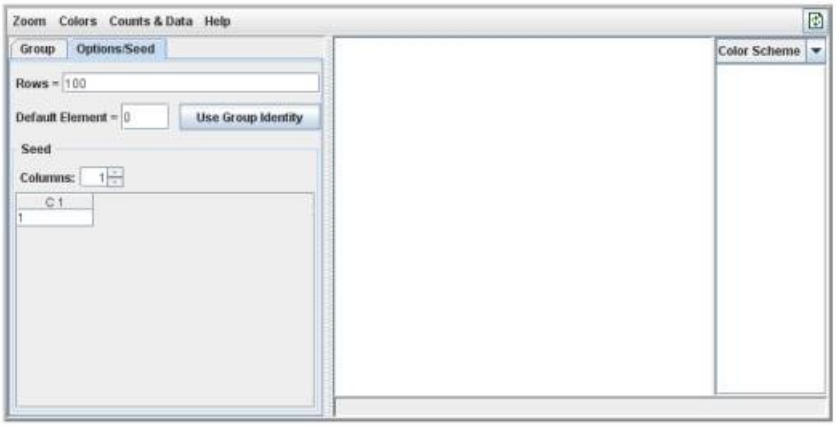
Viewers with Group & Seed Options

There are two applets in this category. The first Viewer with Group & Two Seeds adds the ability to change the group you are working with without changing to a different applet. It also adds the ability to use user-defined groups and the advanced group input. It still uses up to two seed values (you can use one by leaving one of the cells blank), the user can select the number of rows and it has a feature where the user can change the default element.



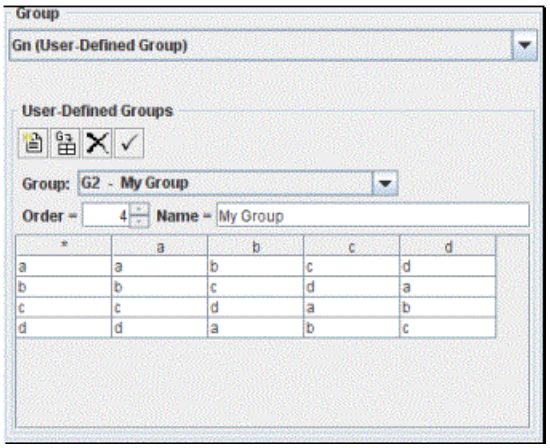
The second applet, Viewer with Group & Seed Table adds the ability to input more than two seed values.





The element syntax for the supported group structures is exactly the same here as in the single group viewers. These two applets add two more options for the group, a user-defined group option and an advanced group input option.

To create a user-defined group select G_n (User-Defined Group) from the group drop-down box. Input the elements of your group down the far left column (these will automatically be copied to the top row) then fill in the rest of the operation table. User-defined structures use the element names that you supply as the element syntax, these are not case sensitive.



Actually, a user-defined group need not be a group at all. The program will only check that the operation table defines a closed operation, all other properties are not checked.

The Advanced input option allows the user to input the group structure using a relatively easy syntax. The advantage to using the advanced input is that you can work with any product or quotient group that uses the built-in group structures or user-defined groups. So if you wanted to create cellular automata over the group $(Z_{10}/\langle 5\rangle \times Q/\langle K\rangle \times D_{10})/\langle (3, J, F2), (2, -I, R3)\rangle$ you can. To use the advanced input option select Advanced from the group drop-down box. You will notice that a text input box appears and probably has Z2 in it. Z2 is the syntax for the group Z_2. We will briefly discuss the group structure syntax below.

Advanced Input Syntax

For any of the built-in group structures the syntax is simply the letter designation of the class of groups followed by the number n (or m) you would have input for that group. For example, the symmetry group of the square (D_4) would be D4, the integers mod 32 would be Z32 and the symmetric group on 5 letters would be S5. For a user-defined group we use a G followed by the number of the group in the list. So G3 would be the third user-defined group in our list of user defined groups. This is also found in the user-defined group drop-down box. In the example above you see that the user-defined group is G2.

Cartesian Products

To take the Cartesian product of groups we simply put an X between the structures we wish to product. So for example, if we want to work with S_3 X D_5 we would use the syntax S3 X D5. We can product more than 2 groups together as well. So to work with D_7 X Q X S_4 X Z_{32} we would use the syntax D7 X Q X S4 X Z32.

As for the element syntax with these groups we simply use ordered tuple notation. For example, if your group is Z3 X Z5 then an element would look like (2, 3). If your group is Z3 X D23 X S7 X Q then an element would look like (2, F10, (1 3 5)(2 4), -J). As with the group syntax, the spaces are not important, except for the elements of Sn where at least one space is needed between each of the numbers in a cycle. A tuple must begin with a (, end with a) and have a single comma between each of the components. One note of caution with products is that tuple simplification is done automatically. That is, if our group is input as Z24 X (S7 X D9) the program will view it as Z24 X S7 X D9 and hence elements should be input in the form (19, (1 2 3), R5) and not (19, ((1 2 3), R5)).

Quotients

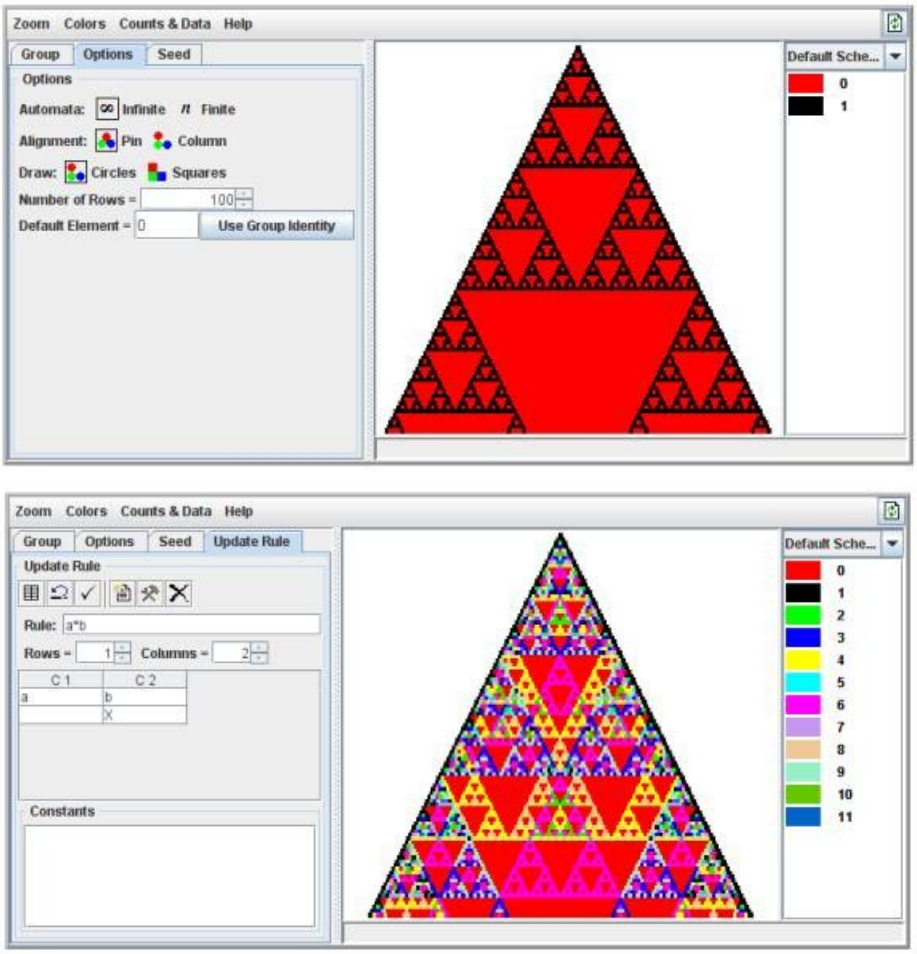
The syntax for quotients is also much like we would write it mathematically. You simply input the group structure followed by a / followed by the subgroup you wish to mod out by. The subgroup is designated by placing the generators for the subgroup between angled brackets <>. So for example, if we want to create automata over the group Z_{10}/<5> we would use the syntax Z10/<5>. If the subgroup has multiple generators we put all of the generators between the angles brackets and separate them with a comma. For example, if we wanted to use S_5 modded out by the subgroup generated by (1 2) and (3 4) we would use the syntax S5/<(1 2), (3 4)>. Note that the subgroup need not be a normal subgroup, the program will simply use the structure of left cosets. If you would prefer to use right cosets instead of left cosets replace the / with a %. So the structure of right cosets of <(1 2), (3 4)> in S_5 would be represented as S5%<(1 2), (3 4)>.

For elements in quotient groups we simply ignore the modded subgroup in the element syntax. For

example, if our group is Z10/<5> an element would be represented as 3 and not as 3 + <5>. There is a printing option for cosets like 3 + <5> but the input parser would not understand the syntax. Similarly, if our group is Z10/<5> X Q an element would look like (3, J). One final example, say our group was (Z10/<5> X Q/<K> X D10)/<(3, J, F2), (2, -I, R3)> then an element would look like (1, J, R1).

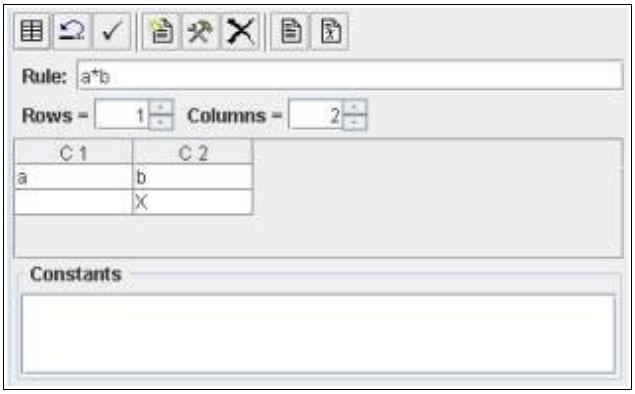
Viewers with Group, Seed & Update Rule Options

There are two applets in this group as well. The first has all of the options of the seed table applet above but adds the ability to graph finite automata as well. The second adds on to this by allowing the user to change the update rule.



Setting the Update Rule

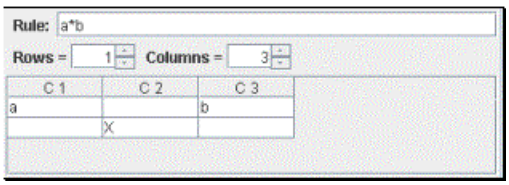
The Update Rule tab looks like,



The last two buttons in the toolbar are for copying information to the clipboard which do not show up in the applet but are present in the PascGalois JE program. The update rule is set, by default, to the Pascal's Triangle update rule. So if you never use an update rule other than Pascal's Triangle you have no need to alter this information. On the other hand, if you wish to experiment with other update rules here is how the updating system is set up.

The rule is any valid group theoretic expression that uses variable names for the group elements and integers for the exponents. Juxtaposition for the group operation is not recognized, you must use a * to denote a group operation. Powers are denoted by ^ and any negative powers must be put in parentheses. Variable names can be any string of letters. Numbers, underscores and special characters are not permitted. For example, the following are legitimate update rule expressions, a*b, a^2*b, b*a*b^(-1), a*b*a^(-1)*b^(-1), and a^3*b^(-4). The following are not legitimate expressions, a^2b and a*b*a^-1*b^-1.

The update grid is how you tell the program where to get the elements that are to be used in the update formula. There must be one and only one X in the entire grid and it must be on the last row of the grid, but it may be in any column. The position of the X tells the program that this is the cell to be updated. The other table entries are names of variables used in the update rule. When the program generates its data it will start at the X and substitute the respective cell entries in for the corresponding variables in the formula to determine the value of the new cell. For example, say we are working with Z_5, infinite automaton, using a column alignment, a seed of just a single 1 and the following update rule.

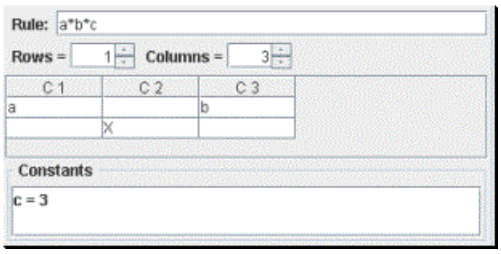


From the grid, when we are updating a cell we will take the element from the last row and one cell to the left and substitute that element in for a in the update formula and we will take the element from the last row and one cell to the right and substitute that element in for b in the update formula. We then evaluate the formula and place the result in the X cell.

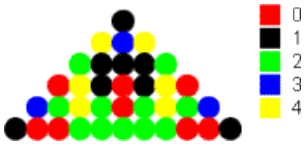
Graphing the first several rows of this automaton gives the following image.

Note that each cell is the sum (mod 5) of the elements to its NW and NE.

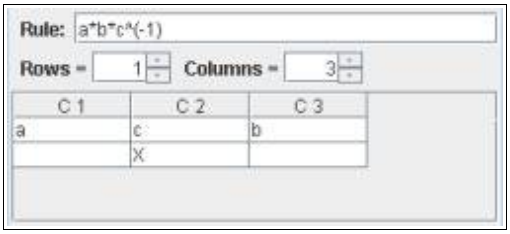
The update rule also allows for the inclusion of constants. Constants must be group elements and are designated in the constants list at the bottom of the tab, not in the update rule expression. To create, delete and edit a constant you use the toolbar buttons, discussed below. For now, let's assume that we have Z_5, infinite automaton, using a column alignment, a seed of just a single 1 and the following update rule.



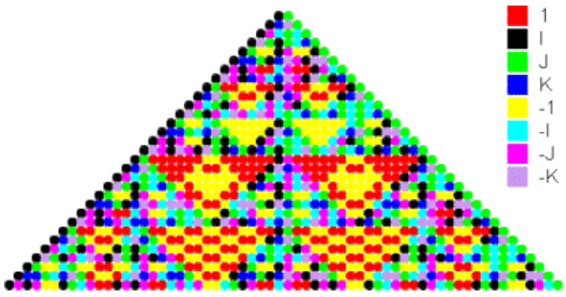
Just as in the last example, each cell is the sum (mod 5) of the elements to its NW and NE but this time we add 3 (mod 5) to that result.



As a last example, say we are working with the Quaternions, default element is the identity, seed is "i j" and has the following update rule.



When graphed, we have the following image.



The Update Rule Toolbar

There are eight tools for the update rule tab. The last two are not present in the applets.

- The first simply clears the update rule grid.
- The second resets the update rule grid to the Pascal's Triangle update rule.
- The third checks the syntax of the update rule as well as the syntax of the constants.
- The fourth tool creates a new constant. When you select this tool the following dialog box appears.

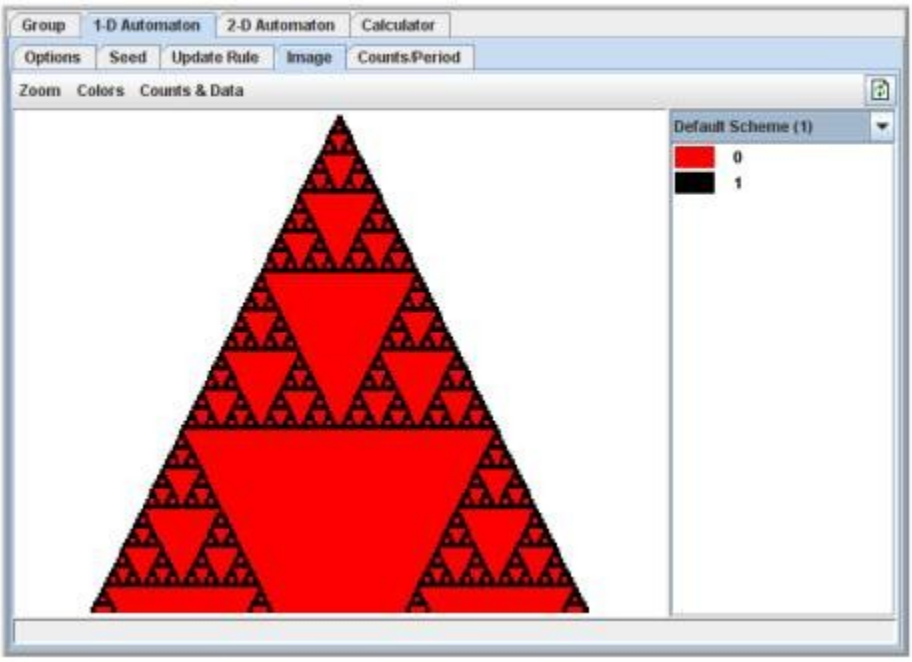
Simply type in a constant name and group element value. The name follows the same restrictions as the variable names, all letters. The value must be a legitimate group element value for whatever group you are currently working with. The syntax of the constant name and the value are not checked when you input or edit a constant. To check the syntax select the syntax checking tool.

- The fifth tool edits a constant. When you select this tool you get the same dialog box as above but the constant name is disabled. Note that you can edit a constant by double-clicking it in the list at the bottom.
- The sixth tool will delete a constant.
- Tools seven and eight copy the update rule formula, grid and constants to the clipboard as either text or LaTeX code.

Viewers with Full Options

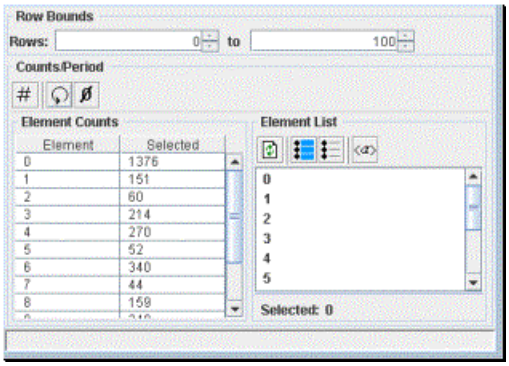
There are three applets in this group. These have almost all of the options that the PascGalois JE application has to offer. Some things that PascGalois JE will do but these applets will not are three-dimensional viewing of two-dimensional automata and advanced counting of sub-triangles and sub-pyramids. The three applets in this group are the 1-D Automaton Viewer, the 2-D Automaton Viewer and the 1-D and 2-D Automaton Viewer. The 1-D Automaton Viewer graphs only one-dimensional automata (like all of the previous applets) but it adds an advanced counting, period and death calculation facility as well as a group calculator. The 2-D Automaton Viewer is for the creation of two-dimensional automata over finite groups. It has many of the same features as the 1-D Automaton Viewer but will graph levels (or time-steps) of two-dimensional automata.

The 1-D and 2-D Automaton Viewer is pictured below. We will discuss the advanced counting options for the 1-D automata here and leave the 2-D automata options for later.



The Counts/Period Tab

If you wish to count the number of elements in a sequence of rows, find the period or death of a finite automaton this tab may be of use to you. The calculations here are independent of those in the image panel so you can count elements in sequences of rows that extend beyond the number of rows you input in the options tab. Furthermore, since the data is regenerated each time a count is done the system does not need to store more rows than what is needed and hence one can do counts on far more rows than would be possible to graph. This tab also has more advanced options for finding the period or death of a finite automaton.



At the top of the tab is the range of rows to be used. This range is inclusive. Directly below that is the counts table and element list, similar to the setup of the counts dialog box that is displayed when the display counts option is invoked from the image tab. The selected column is, of course, the number of the elements counted in the range of rows.

The Counts/Period Toolbar

The Counts/Period Toolbar consists of a single option (three options for finite automata).

- The first generates the data and displays the new counts in the table.
- When the automaton is finite there will be a second option to find the period. The period calculation here is a bit different from the period calculation in the image tab. In the image tab the program will start with row 0 and try to find the period starting at row 0. If this is not found it will try to find the period by starting at row 1, and then row 2 and so on. So if a period is in the graphed set it will be found. In the counts/period tab the period is found a bit differently. The program will try only once starting at the beginning row. So it is possible for the image tab to find the period and the counts tab not to, if the beginning row is not set high enough. What you want to do is set the beginning row fairly large, possibly larger than the number of rows you have set in the options tab and then set the ending row very large.
- When the automaton is finite there will also be a third option to find the death of the automaton. The calculation of the automaton death searches for a row (or set of n rows if the depth of the update rule is n) that match the group identity. If a set of identity rows are found the program will tell you where the death starts. That is, the last row number of the first block of identity rows that are the same depth as the update rule.

The Element List Toolbar

The element list toolbar has four simple options.

- The first option refreshes the element list. Theoretically you should never need this option but as with the image tab the calculations are done in separate threads and if the element list for some reason does not appear this option will display it.
- The second option selects all of the elements in the list.
- The third option unselects all of the elements in the list.
- The fourth option selects the subgroup generated by the currently selected elements.

This concludes the introduction to the PascGalois applet system. The PascGalois JE program has a very similar interface to that of the latter applets and it has all of the facilities as the applets with full options, plus a few more. At this point it should be clear how to use the 1-D automata features of the PascGalois JE program. The main additions that the PascGalois JE program offers are file saving and loading of automata information, saving of images, and the clipboard transfer of images and information.